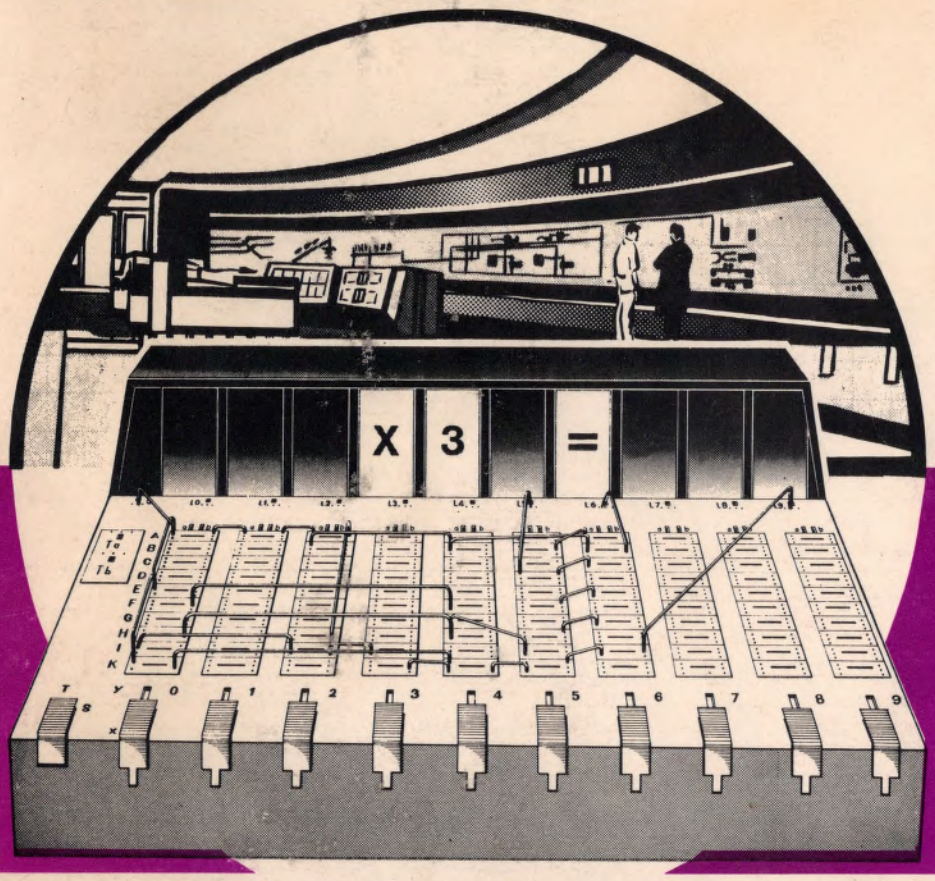
Science Fair®

SF-5000 ELECTRONIC COMPUTER

BY ROLF LOHBERG



PROGRAMMING MANUAL

AND ASSEMBLY INSTRUCTIONS

ENGINEERED FOR

SCIENCE FAIR ELECTRONICS A TANDY CORPORATION COMPANY

SCIENCE FAIR ®
SF - 5000 ELECTRONIC COMPUTER

PROGRAMMING MANUAL
& ASSEMBLY INSTRUCTIONS

Copyright © 1971
by
FRANCKH'SCHE VERLAGSHANDLUNG, KOSMOS
Stuttgart, West Germany

This book, or any part thereof, may not be reproduced by any means without the prior written permission of the publisher.

3rd. English Edition.

Printed in Canada

Preface

Electronic calculators have greatly influenced our way of life. These days it is impossible to understand modern science or technology without knowing something about how these electronic devices work. With the Miniature Computer you can learn the basic principles of modern computer-science and at the same time enjoy many interesting hours, assembling your Miniature Computer and programming it to solve problems.

You will work step by step, first with very simple programs, then with more and more complex ones; eventually you will be doing calculations in binary arithmetic with your Miniature Computer. The same system is used by most real computers. In this very interesting and entertaining way, you will be gaining a head start in understand computer-science, so important in practically every aspect of modern life.

TABLE OF CONTENTS

Preface 1

I. How to assemble your Miniature Computer 5

II. Getting to know your Miniature Computer 11

III. Programming your Miniature Computer 15

Simple Programs
- Program 1 - Electrical Test 16
- Program 2 - A Flashing Sign 16
- Program 3 - The Computer Casino 17
- Program 4 - Hide And Seek 20
- Program 5 - Crossing The River 21
- Program 6 - To Catch A Robber 22
- Program 7 - The Computer Doctor 23

Logic
- Program 8 - Passenger Booking With The "AND Circuit" 26
- Program 9 - Passenger Booking With The "OR Circuit" 27
- Program 10 - What Will The Weather Be Like? 31
- Program 11 - Bookkeeping 33
- Program 12 - The Computer Barman 34
- Program 13 - Student Time-Table 34
- Program 14 - From Colors To Logic 35
- Program 15 - Primary, Secondary And Tertiary Colors 37
- Program 16 - Paper, Stone, Scissors 38
- Program 17 - A Knowledge Test 38
- Program 18 - Sorry, You Lost! 40
- Program 19 - Cat And Mouse 41
- Program 20 - A Translating Machine 42
- Program 21 - The Computer Detective 43
- Program 22 - Telephone Switchboard 48

Cybernetics
- Program 23 - A Warm Bath 53
- Program 24 - For A Happy Fish 55
- Program 25 - Body Heat 57
- Program 26 - The Human Eye 57
- Program 27 - Heating A House 58
- Program 28 - How A Dog Learns 60

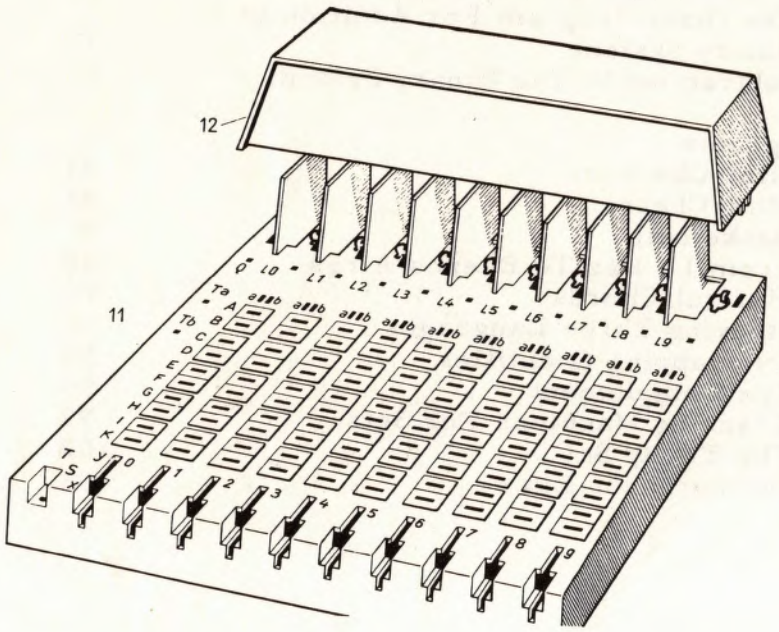
Mathematical Calculations
- Program 29 - The Calculator 62
- Program 30 - Addition By Selection 65
- Program 31 - Counting 66
- Program 32 - Mathematical Squares 67

Program 33 - The Binary System 69
Program 34 - Binary To Decimal Conversion 71
Program 35 - Division Program 72
Program 36 - How To Calculate With O And L 73
Program 37 - A Two-Place Binary Adder 76
Program 38 - The Giant Program For Addition In The Binary System 77
Program 39 - Subtraction In The Binary System 81

More Advanced Programs
Program 40 - Mini-Checkers 83
Program 41 - Mini-Chess 84
Program 42 - Basketball 87
Program 43 - From La Paz To Buenos Aires 88
Program 44 - Musical Chords 89
Program 45 - Studying Three Languages 90
Program 46 - Overlapping Quantities 92
Program 47 - The Complement 95
Program 48 - Teaching Machines For Sets 99
Program 49 - The Empty Set 100
Program 50 - De Morgan's Laws 103

MINIATURE COMPUTER

List of Parts



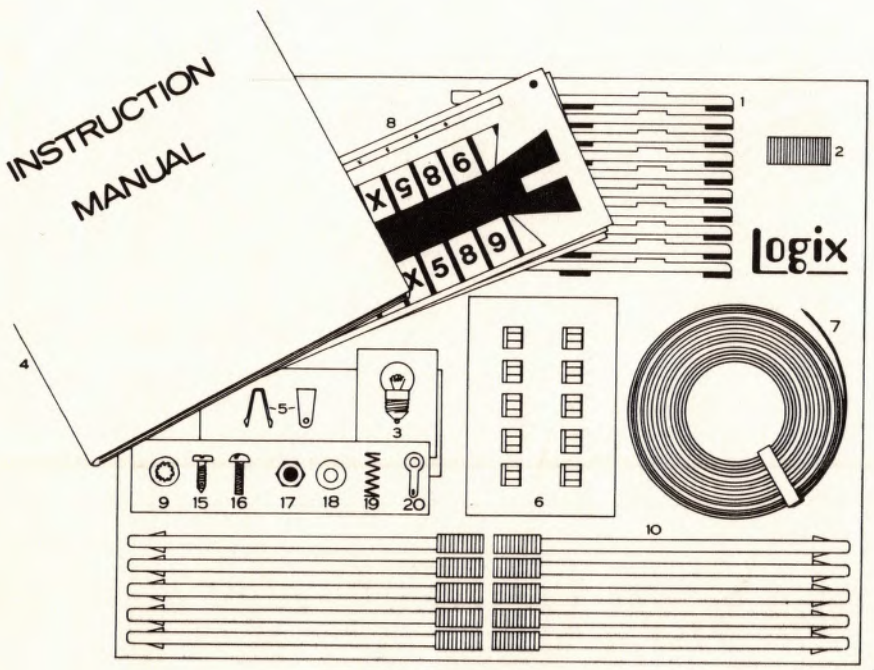


FIGURE 1

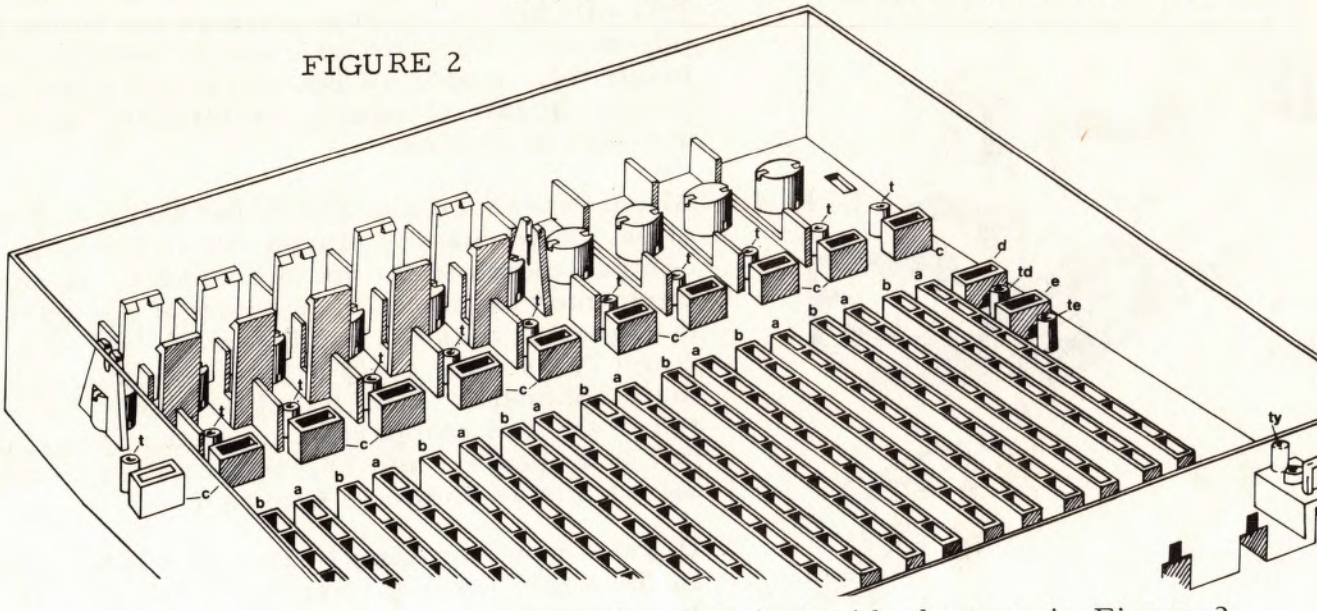
1. - Plastic clamps (10)
2. - Push-button
3. - Light bulbs (10)
5. - U shaped brass clips (50)
6. - Three pronged brass clips (213)
7. - Programming wire
8. - Transparent sheets (42)
9. - Lock washers (2)
10. - Red sliders (10)
11. - Main console
12. - Indicator lamp housing
15. - Mounting screws (14)
16. - Battery terminal bolts (2)
17. - Hexagon nuts (4)
18. - Washer
19. - Compression springs (2)
20. - One eyed clips (14)

HOW TO ASSEMBLE YOUR MINIATURE COMPUTER

( ) 1. Figure 1 is an illustration of all the parts that are used in assembling your Miniature Computer. Check the parts included in your kit against the list of parts in Figure 1.

( ) 2. When you examine the console, you will notice ten slots at the lower edge of the console labelled S0, 1, 2, 3, ...9. The red sliders should be inserted in these slots as follows:

FIGURE 2



Turn the console upside down as in Figure 2. Take a red slider and fit it to the console by pushing the slider through the plastic channel between "a" and "b" on switch S0 (see Figure 3). The red slider should be held down firmly inside this channel while it is being pushed in.

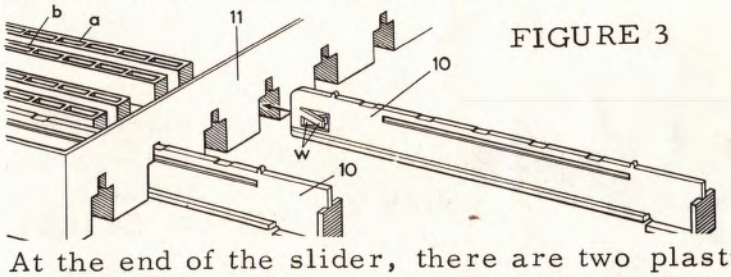


FIGURE 3

At the end of the slider, there are two plastic prongs ("w" in Figure 4) which rub against the sides of the channel during insertion. The slider must be pushed in fully. When the slider prongs have passed through the end of the channel, the prongs will spring out and will now hold the slider permanently in the channel.

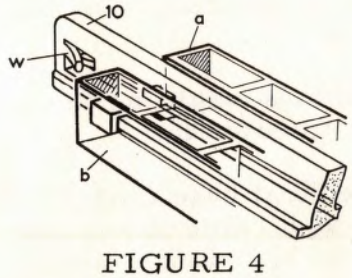
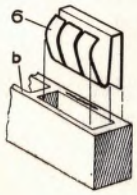


FIGURE 4

Assemble all ten sliders to the console in the same way.



( ) 3. Take the bag containing the three-pronged clips (Part no. 6). One of these clips fits into each o the rectangular compartments marked "a" and "b", as shown in Figure 5.

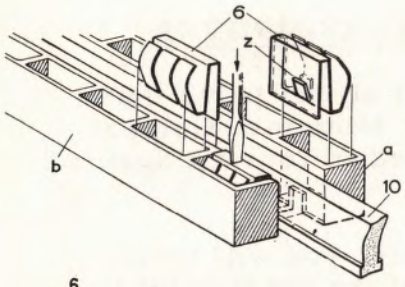


FIGURE 5

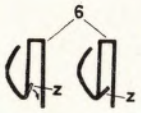


FIGURE 6

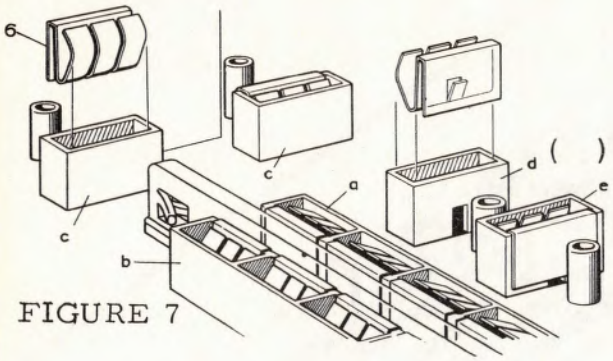
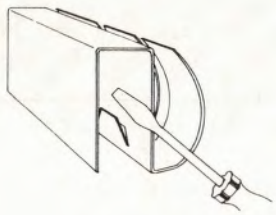


FIGURE 7

Insert each clip into a rectangular compartment, pushing the clip carefully with the fingers and then using a screwdriver to tap the clip lightly. The three prongs fit inside the compartment; the flat plate of the clip fits on the outside wall of the compartment facing the slider. At the bottom of the clip is a tiny metal prong (marked "z" on Figures 5 and 6), which should click into the hole specially provided for it in the holder. You will hear a distinct click as the prong springs into place. The clip will now be held there permanently. Insert all 200 clips in the rectangular compartments alongside the sliders, in the manner just described.

( ) 4. Altogether there are 213 of the three-pronged clips. After having placed 200 of the clips in the compartments in the main channels, 11 additional clips must be placed in the row of compartments that goes across the top (marked "C" in Figures 2 and 7), so that their prongs face the row of main channels. Finally, clips should be inserted in the two compartments at the side ("d" and "e" in Figures 2 and 7), so they face in opposite directions as shown in Figure 7.



If while programming (done at a later stage in the book), a clip happens to fall out, it can be secured in place as follows: Using a small screwdriver or a pocket knife, open the small prong "z" as shown in the diagrams on the left. Re-insert the clip which should now hold firmly in place.

Cut off a piece of wire (No. 7) about 20 inches in length. Strip the wire bare by removing the plastic insulation (covering).

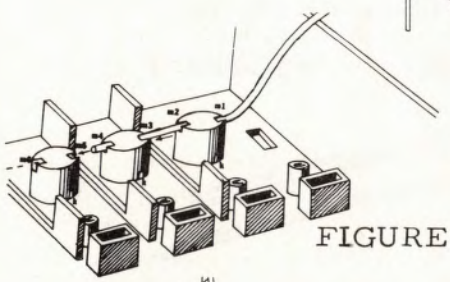


FIGURE 8a

( ) 5. Starting at the right hand end, under light socket LO, we want to thread the wire in a line straight through the two holes (m) at the bottom of each of the light sockets (Figure 8a).

An easy way to accomplish this would be to use a "sewing" motion as described below:

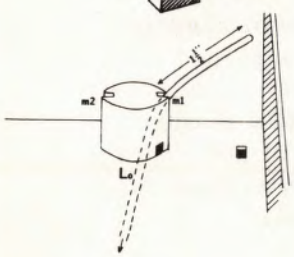


FIGURE 8b

Insert the wire through the first hole (ml) in the light socket LO, from the underside to the top-side of the console (Figure 8b). Leave about 1 1/2 inches of wire extending from the bottom of this socket so that the wire can be anchored.

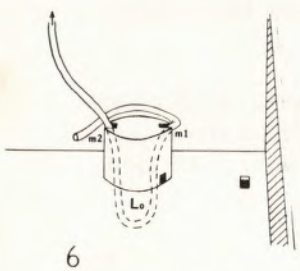


FIGURE 8c

Bend this 1 1/2 inch end of wire around the light socket LO in order to prevent it slipping through the hole "ml" (Figure 8c). Continue the sewing motion by passing the free end through the second hole "m2" in the socket LO, so that the end of the wire again emerges on the bottom side of the console (Figure 8c).

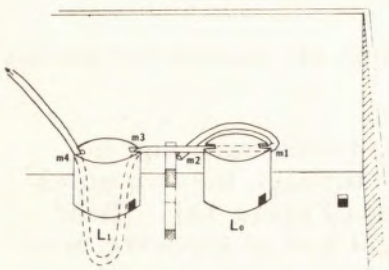


FIGURE 8d

Pull the wire taut so that it lies in the groove at the bottom of the socket LO. (You can see this clearly by looking inside the socket from the top-side of the console).

Repeat the sewing motion for socket L1 (See Figure 8d).

Continue this sewing motion until the wire has passed through all ten light sockets. Care should be taken to avoid kinks in the wire and to keep the wire straight, especially inside the light sockets between the holes (m1 and m2, m3 and m4, m5 and m6, etc.). The wire should lie in the groove or channel at the bottom of the inside of each of the light sockets.

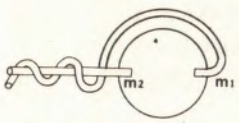
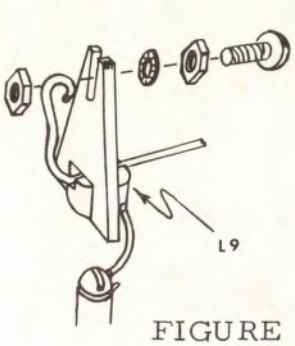


FIGURE 9

To anchor the wire firmly at the light socket LO, twist the wire as shown in Figure 9.

'EXPLODED' VIEW END VIEW



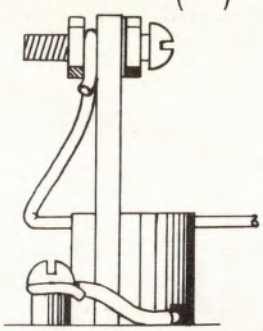


FIGURE 10

( ) 6. At the left hand end of the bare wire (at L9) form a small loop.

Take a bolt (Part no. 16) and mount a nut (no. 17) on it. Place a lockwasher (no. 9) on the bolt next to the nut (see Figure 10). Pass the end of the bolt through the slot in the battery terminal post (at light socket L9) and hook the loop of the bare wire on the end of the bolt. Mount another nut (17) on the bolt and tighten the assembly.

( ) 7. Cut a 6 inch length of wire. Strip 1/2 inch of insulation off both ends and curl the bare ends. This can be done easily by bending each end around the shaft of a screwdriver, as shown in Figure 11.

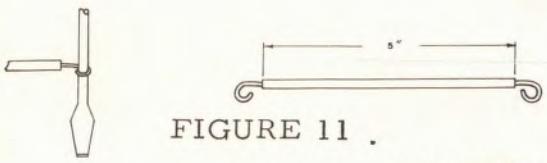


FIGURE 11.

'EXPLODED' VIEW END VIEW

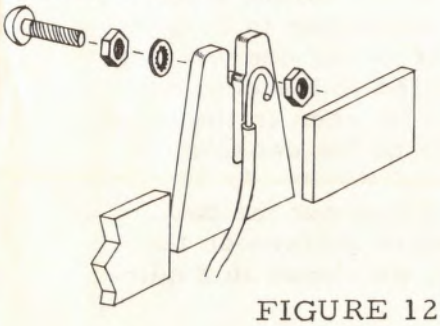


FIGURE 12

Connect one end of the wire to the second battery terminal post (located between light sockets L3 and L4) using a bolt (16), two nuts (17) and a lockwasher (9) as shown in Figure 12. Use the same method of assembly as in Step 6 above.

Note: That in Steps 6 and 7, the heads of the bolts should be inside the battery compartment so that the batteries will make contact with the circuit by touching the heads of the bolts.

( ) 8. Cut off 10 strips of wire, each 2 inches in length. Taking each length of wire in turn, strip off 3/4 inch of the insulation from each end.

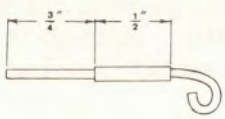


FIGURE 13

Curl <u>one</u> end of each wire as explained in Figure 11. Note that the second end of each wire will remain straight as shown in Figure 13.

FIGURE 14

( ) 9. Perform the following steps on each of the pieces of wire:

a) Make sure that the 3/4 inch bare end is straight and insert it through the two holes (k) in the light socket (Figure 14). Bend the protruding straight end of the wire back around the light socket to prevent it from slipping out of the light socket holes.

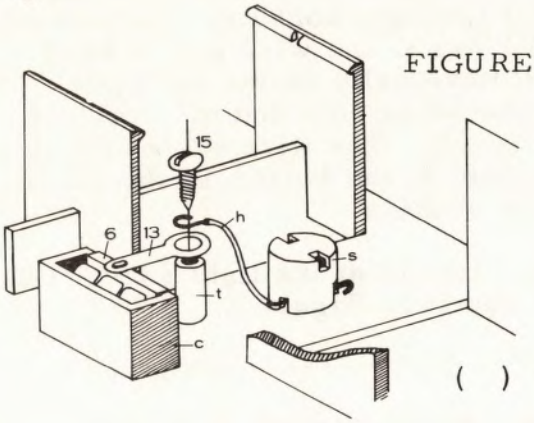


FIGURE 15

b) Connect the curled end of this wire to the round pillar "t", at the same time mounting a one-eyed clip onto the pillar and using a screw (15). Make sure that the finger of the clip touches the top of the three-pronged metal clip (Figure 15)

( ) 10. Connect the free end of the insulated wire, mentioned in Step 7 above, to the round pillar 't' at the extreme right-hand end of the row of eleven pillars (Figure 16).

HINT

right wrong

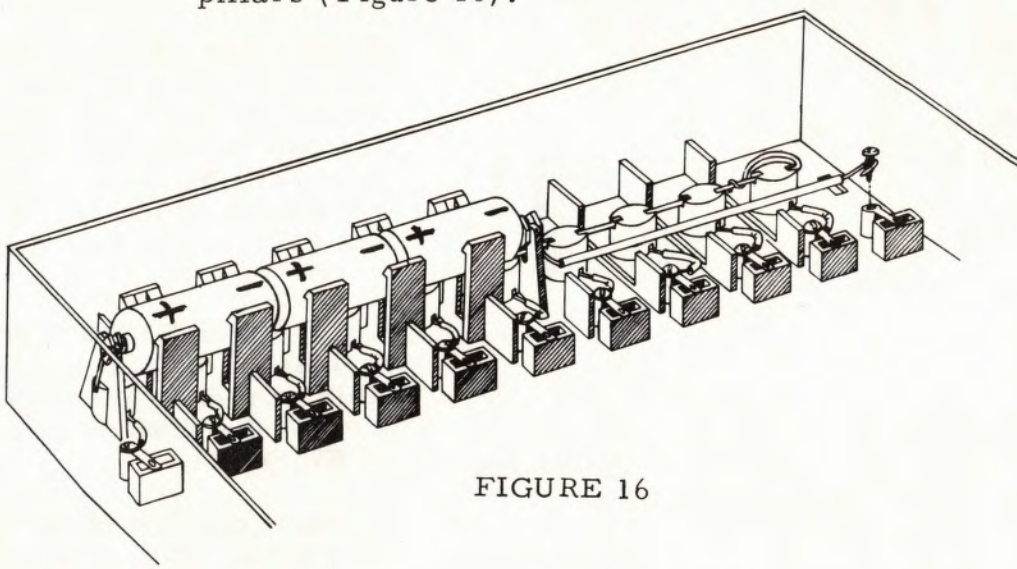


FIGURE 16

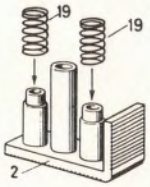


FIGURE 17

When you are connecting the wire, mount a one-eyed clip and use a screw as explained in Step 9 (b) above. Note that the pillar is not connected to any bulb. It is instead connected to the common power terminal, Q, which can be seen in the upper left-hand corner of the topside of the console.

( ) 11. Now install the push button. First put the two small springs (19) over the short pillars on the bottom of the push button (2), as shown in Figure 17.

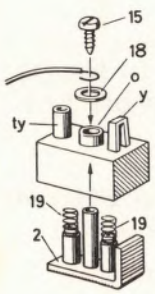


FIGURE 18

Take a piece of wire 7 inches long and strip 1/2 inch of the insulation off each end. Now push the button together with the springs into its housing on the left side of the console. Attach it permanently by putting a mounting screw (15) and a

washer (18) into the center hole in the housing (Figure 18).

Before tightening the screw, wrap one end of the wire around it so that it is firmly fastened to this part of the housing. Tighten it so that the button can still be pushed in on the springs.

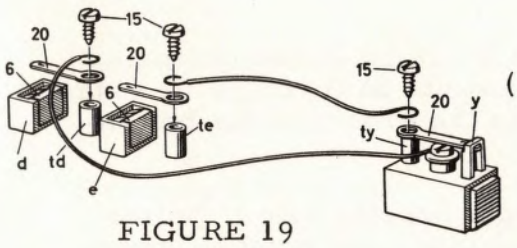


FIGURE 19

( ) 12. Use a screw and a one-eyed clip to attach the other end of the wire (from the push button) to the further of the two rectangular housings ("d" in Figures 2 and 19). Make sure that the finger of the one-eyed clip touches the three-pronged clip in the compartment (Figure 19).

( ) 13. Take another one-eyed clip and screw it into position at "ty" in Figure 19 (also shown in Figure 18), first slipping the tip of its finger through the gap "y" at the foot of the push button housing.

Take another piece of wire 6 1/2 inches long, strip the insulation off both ends and attach one end to the screw holding the clip in position; the other end of the wire should be connected to the housing "e" through a one-eyed clip at te, as in Figure 19.

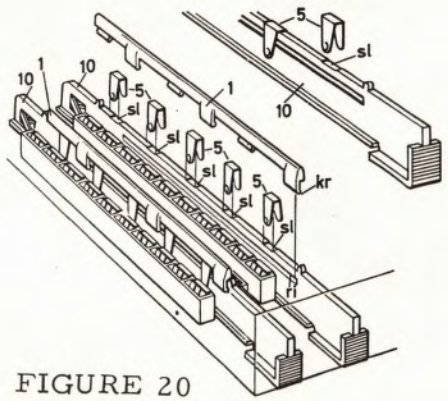


FIGURE 20

( ) 14. Affix the U-shaped clips (5) to the bottom of the red sliders. Each slider has five small indentations (marked sl in Figure 20).

One brass clip fits in at each indentation. Hold the U-clips in place by using the long plastic clamps (1). Note that these clamps have three prongs (marked "kr" in Figure 20) on each side. These prongs snap into position on both sides of the red sliders, fitting into ridges "ri", and hold the brass clips in place.

Note: Do not squeeze or deform the U-clips excessively since they must retain their shape and spring in order to maintain good electrical contact.

( ) 15. Finally, snap the batteries into their housing as shown in Figure 16. Be sure that the negative and positive terminals of the battery cells are pointed as shown.

Since three "C" cells must fit into the battery holder between the terminal posts and since the lengths of batteries vary slightly, it may be necessary to adjust the space between the terminal bolt heads to allow a snug fit and good electrical contact between the batteries and the bolt heads.

If there is too much space between the terminals, turn one of the bolts two times in a counter-clockwise direction. Repeat this procedure if the batteries still do not fit snugly.

If, on the other hand, the batteries do not have enough room to fit in, turn one of the bolts two times in a clockwise direction. It may be necessary to remove the inner nut altogether to provide adequate space for the batteries.

( ) 16. Turn the main console over and screw one bulb into each of the light sockets until contact is established with the wire in the groove at the bottom of the socket.

Carefully put on the light housing (12). Notice that it has two different sides and must be mounted properly. You will have to remove the light housing to change the transparent sheet for each program. Be sure that you cut and fold the transparent sheet carefully (Figure 21).

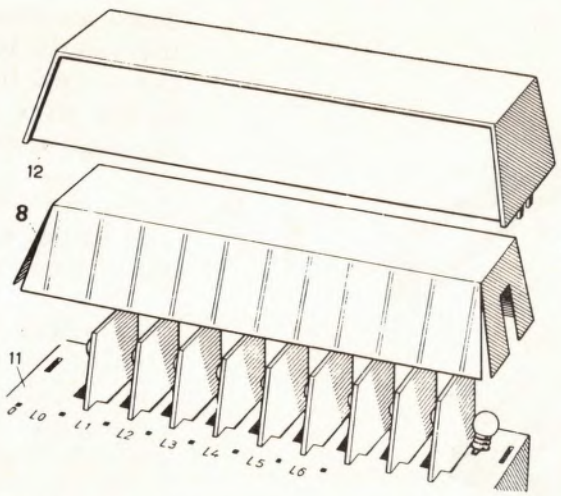


FIGURE 21

There are some programs that require the additional strip along the bottom: be sure to insert this carefully (Figure 22).

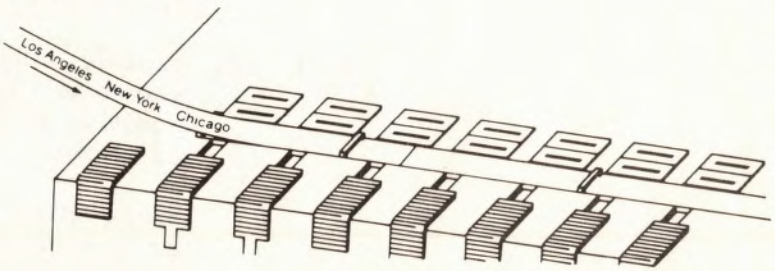


FIGURE 22

II. - GETTING TO KNOW YOUR MINIATURE COMPUTER

Whether or not you have assembled your Miniature Computer yourself, you should take a little time now to become familiar with its working parts. When you begin the experiments and problems, this will save you many errors and wasted time.

The figures on the opposite page show you two important things. First, the schematic drawing shows you how we are going to represent the parts of your computer in our wiring schemes. We are going to be using many of these figures. They will not be too hard to follow if you start out by carefully comparing the two drawings.

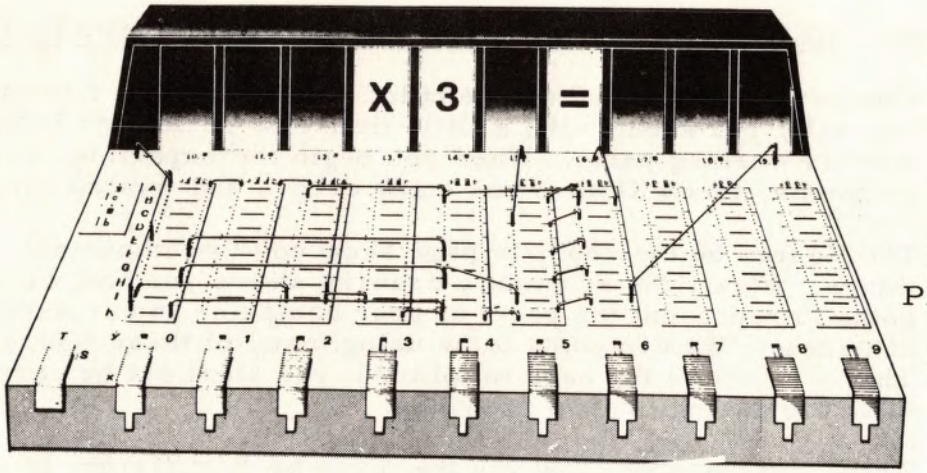
These figures also show you the three basic divisions of your Miniature Computer:

1) The Input Section
2) The Program or Logic Section
3) The Output Section

Any calculating device has three basic sections. A mechanical desk adding machine, one that can add and subtract, for example, must have at least three basic parts: there must be a way to put the facts of the problem into the calculator; there must be a section that makes the calculations; the answer must somehow get back to the operator in a comprehensible manner. Let's expand this concept a little: The first task, putting the facts of the problem into the calculator, is accomplished in the adding machine by punching keys, which puts the facts (digits) of the problem into the machine. This device, which puts the facts into the adding machine, is called the Input. Your Miniature Computer also has an Input. Instead of putting in the facts of the problem by punching keys, as in the adding machine, you use the ten sliders which together make up the Input Section (Section 1) of your Miniature Computer.

It is obvious that to solve a problem with the adding machine, once you have punched in the figures, the machine must have a way of using them so that the correct answer is found. This is done mechanically through a series of gears. In a computer, this area is known as the Logic Section, since it carries out logical or mathematical manipulations of the digits (or other data) that have been fed in. In your Miniature Computer, the logic work is done through the switches that have already been assembled, as well as by the wiring you will put on the machine during each experiment.

This section also serves another purpose. Remember that the simple desk adding machine we have been talking about, can do two operations: addition and subtraction. We only have to press down a special control key on the machine to change it from an "adding machine" into a "subtraction machine". This is the act of programming the operations that the machine will carry out. With your Miniature Computer, you program the form of its operations by wiring it in different ways. Thus, you could say that you are the real control unit of your Miniature Computer.

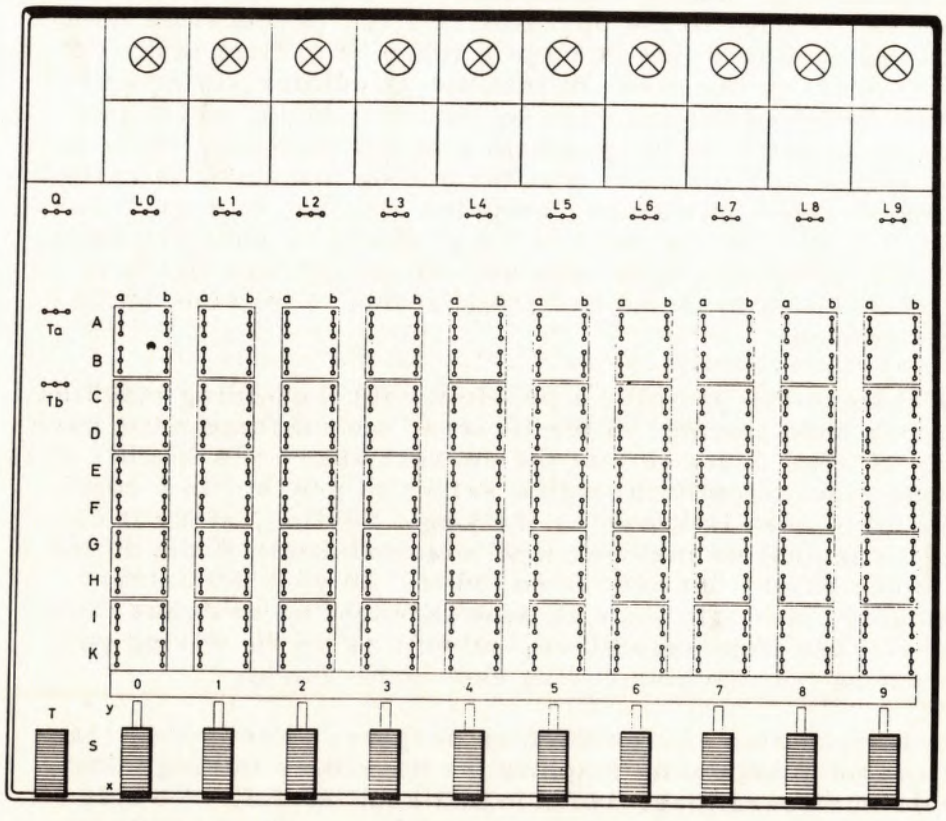


SECTION III
OUTPUT

SECTION II
PROGRAM - LOGIC

SECTION I
INPUT

ILLUSTRATIVE DRAWING
(as you see your miniature computer)



SECTION III
OUTPUT

SECTION II
PROGRAM - LOGIC

SECTION I
INPUT

SCHEMATIC DRAWING
(as used in this book to clearly show your computer)

Finally, any calculating device must be able to express the results of its work in an answer that the operator can use in some way The desk adding machine does this by printing the results of its work on a strip of paper. Your Miniature Computer, however, signals its answer by lighting up one or more (or sometimes none) of the ten bulbs in Section 3. This is known as the Output operation since, just as the data was "put in" in Section 1, it is now being "put out".

REAL COMPUTERS WORK THIS WAY TOO!

Giant computers that carry out complex operations in a thousandth of a second, also have Input, Programming, Logic and Output Sections; many thousands of times more complex than the ones in your Miniature Computer.

INPUT

There are many forms of computer input used today. One of the earliest was the common punched card. Punched paper rolls are also used. A faster method of input is through magnetic tape, like that used in tape recorders. Now, of course, there are even "optical readers" which can read the coded magnetic numbers on checks and speed up banking operations. In a large computer, the Programming and Logic Sections are separate. There is also a fixed section dealing with "memory" and "storage".

PROGRAMMING

The Programming Section is also sometimes called the "Control" Section, and this is what it does exactly. As we mentioned in the introduction, computers can carry out in minutes fantastically complex calculations - ones that would take centuries for the unaided human mind. However, at the same time, the computer is incredibly simple-minded. In order for it to be able to digest a problem, it must be fed in little bite-sized chunks; this means that the problem and the simplified steps for its solution must be prepared in advance and fed into the computer step by step. Once it has been given the methods, the computer will follow the directions with lightning speed and quickly find the solution. It is a bit like the mechanical monster robot of the movies. It has immense powers; but if it is to work properly, it must be told by its creator what to do at every step. In a computer, forming and sending these instructions is known as programming. You program your Miniature Computer by wiring it in such a way that it will perform the desired action. In a large computer, instructions are fed into the calculator in machine code and it is programmed automatically.

LOGIC

The Logic Section of nearly all computers, even the most

advanced ones, works basically in a very simple way. This is the Binary or "OFF - ON" system. We can use the example of a light bulb.

You want to send a secret signal to a friend who lives next door. You tell him to watch the window of your room: if, let's say, the light is on at six o'clock, he should come over; if the light is off, he shouldn't. Thus, you have been able to send a message through this simple two-position technique. Later on in your experiments with your Miniature Computer, you will learn how large computers make use of this technique to show figures and facts in a method perfectly suited to the computer's miniature electronic circuits.

LOGIC

Large computers have a separate Memory Section. In a simple way, our Miniature Computer also has a memory; in fact when we feed a certain piece of information into it and do not change anything else, the fact will not disappear or dissolve. However, real computers have separate memory units in which huge amounts of data may be "stored" almost indefinitely, and the real computer can use this stored information whenever it is directed to do so

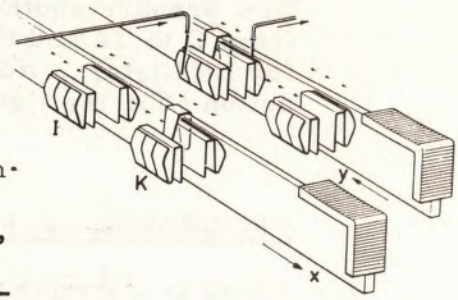
OUTPUT

Finally we come to the Output Section. Remember that your Miniature Computer puts forth its output by lighting up a pattern of light bulbs. In large computers, output may take many forms. Punched cards or tapes are the most common. Some computers print their answers on sheets of paper so fast that a page of a book can be produced in about a second. Other forms are electronic beams on screens that show shapes and patterns, electronic signals and even sounds.

III. - PROGRAMMING YOUR MINIATURE COMPUTER

Now that you know something about the structure of your Miniature Computer, you can start to work with it.

Remember that the Input Section consists of the ten sliders. Any slider can be in either of two positions: forwards (y) or backwards (x). (You will see the letters x and y clearly imprinted on the Programming Board of your computer.)



Turn your computer over and notice what happens underneath. Notice that by running the slider from x to y, you are shifting a series of connections on the Programming Board from B, D, F, H, K to A, C, E, G, I. Thus, any wires connected to both the in (a) and the out (b) terminals of Switch I, will transmit current only when the slider for that row is in position y. Wires connected to Switch K, will only be connected when the slider is in the x position. This will become easy to understand as you practice with your Miniature Computer.

The Programming Board contains 600 tiny holes. At first, you program by connecting holes on the Programming Board with wires according to our diagrams. Later, when you have gained experience, you will invent your own programs.

Note that the holes in the Programming Board come in groups of three. Each member of a group of three is identical as far as we are concerned. When a wire is to be connected to 2Cb for example (that is vertical row 2, horizontal row C, on the right side), we can use any of the three holes at that position.

When preparing a wire to connect two holes, cut off a length of about an inch longer than necessary. Strip off about 3/8 of an inch of insulation from each end. Bend the wire at a right-angle just above the stripped part at either end. This will prevent the insulation from sliding over the stripped end.

The Output Section consists of ten numbered compartments. Each of these contains a bulb. For most programs, there is a transparent sheet that you can cut out (carefully) and fit into the Output Section, as well as a thin strip for the Input Section.

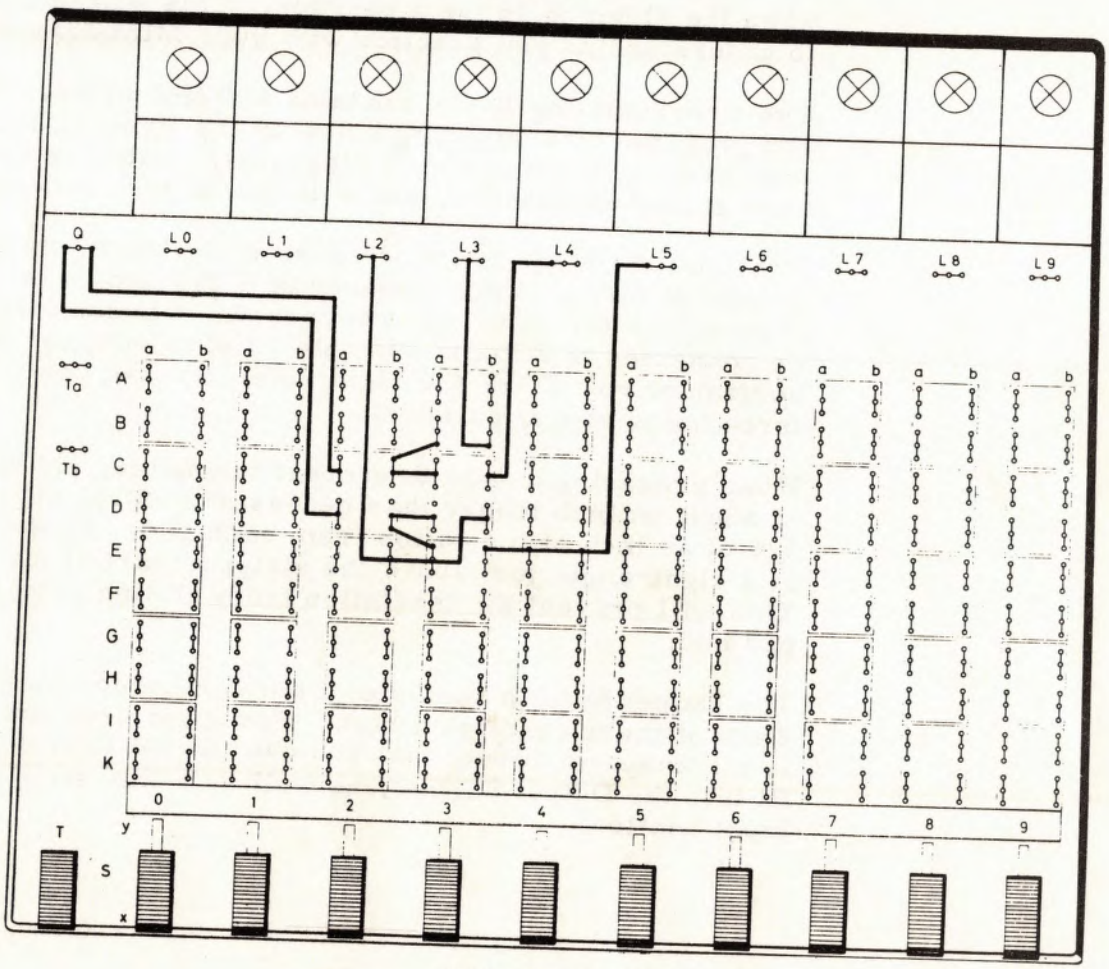
PROGRAM 1 - ELECTRICAL TEST

Prepare a strip of wire about 12 inches long, strip off the ends and bend them as explained in the last section. Put one end of the wire into one of the three holes at Q and put the other end into any of the holes located directly in front of the Output Section - that is, on the same row as Q. A bulb should light up as you do this. Test all the bulbs in this way.

Now prepare another length of wire. Connect one length from Q to Ta and the second from Tb to one of the holes for L (Light). Nothing happens. Now press on the push button (T): the light goes on!

PROGRAM 2 - A FLASHING SIGN

There is a simple way to get an exciting effect with your Miniature Computer - something like a flashing, electric sign. Lead two wires from Q to 2Ca and 2Da. Follow the rest of the wiring scheme exactly.



WIRING SCHEME 2

Now, alternately move Sliders S2 and S3 to y. When S2 is at y, S3 is at x - and so forth. If you keep switching fairly

rapidly, a light will seem to dance back and forth in the Output Section above.

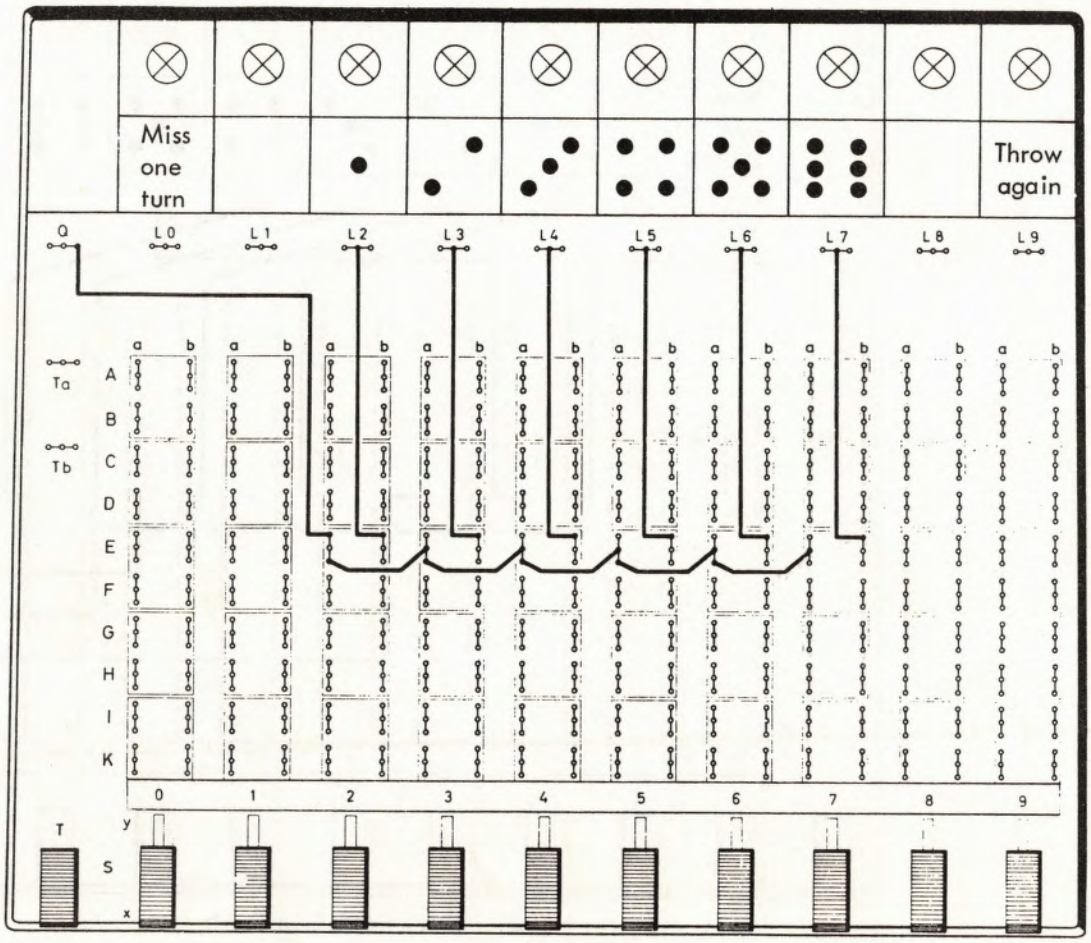
Study the wiring in this simple program carefully. It will make the more complicated wiring schemes that follow later much easier to understand.

HINT 1: Wiring becomes easier when you are using different colored pencils. As you put each wire in place on the programming board, follow it on the diagram with the pencil.

PROGRAM 3 - THE COMPUTER CASINO

- Use transparent sheet No. 3.

We cannot teach our Miniature Computer to spin a roulette wheel, but we can teach it at least three interesting ways to play a game of dice. The first one is very simple. By the time you set up the third game, our computer will be able to challenge even a sharp player.

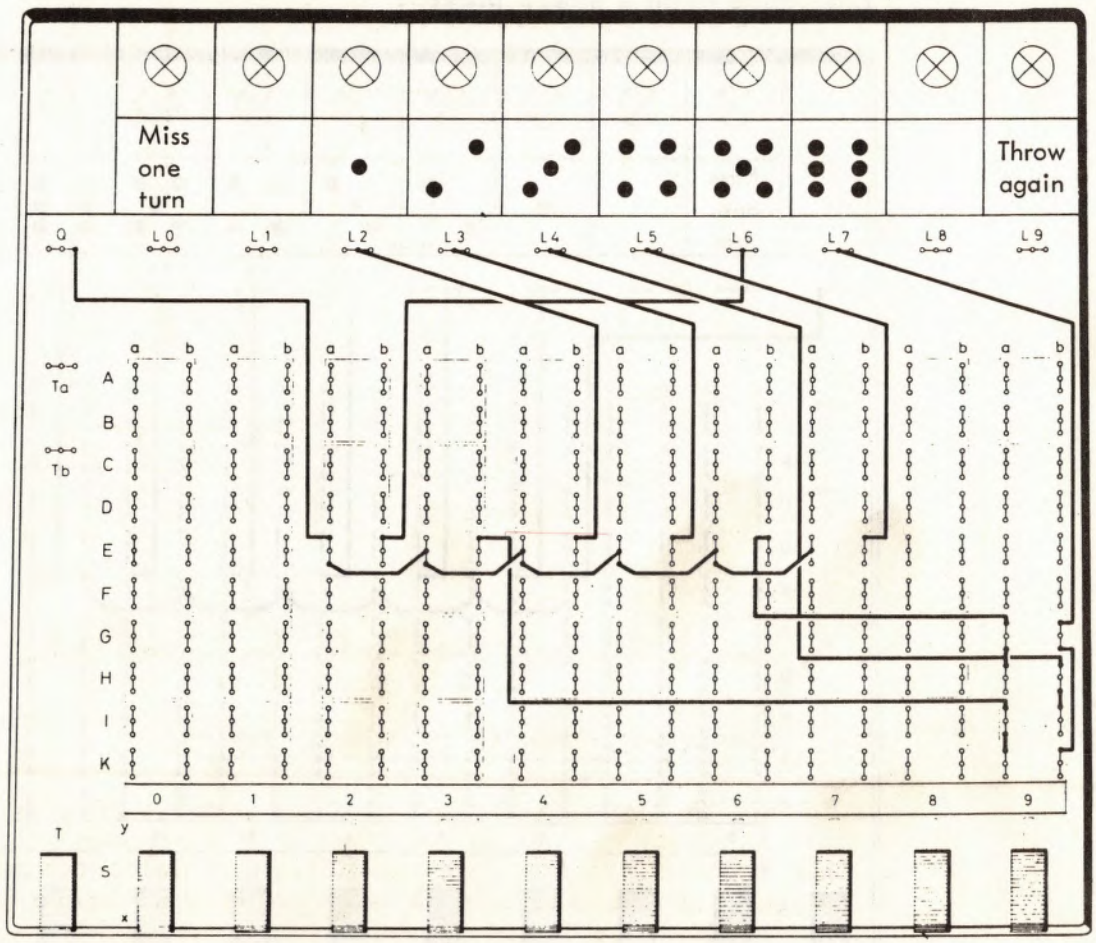


WIRING SCHEME 3a

Wiring Scheme 3a shows you our first variation. It is very easy to wire, but notice carefully how we have connected the

current from Q to the six switching elements, Sliders 2 to 7. We cannot run individual wires from Q to each of the six. This would require too many outlets at Q and too much wiring. Therefore, we have simplified it. Only 2E is connected to Q. From 2E another wire leapfrogs to 3E, and from 3E another wire jumps to 4E. This continues until the current reaches 7E without any unnecessary connections. The designers of large electronic computers work hard at simplifying all connections as much as possible. After all, sixty or seventy thousand circuits and connections are complicated enough! Unnecessary connections would make things impossible.

Now that our wiring is finished, we can start our dice game. Find a partner to play with - preferably not a computer expert. Move any one of the sliders between S2 and S7 from x to y and the value of the dice will light up. Next, it is your partner's turn. Exciting? Yes, but only for a while. It won't take too long for your partner to figure out how to throw "six" - even if he isn't a computer expert. So let's try to make things a bit more complicated. If you rewire according to Wiring Scheme 3b, our dice game will offer more of a challenge. This time, it will be a little more difficult to figure out which slider will produce a "six".



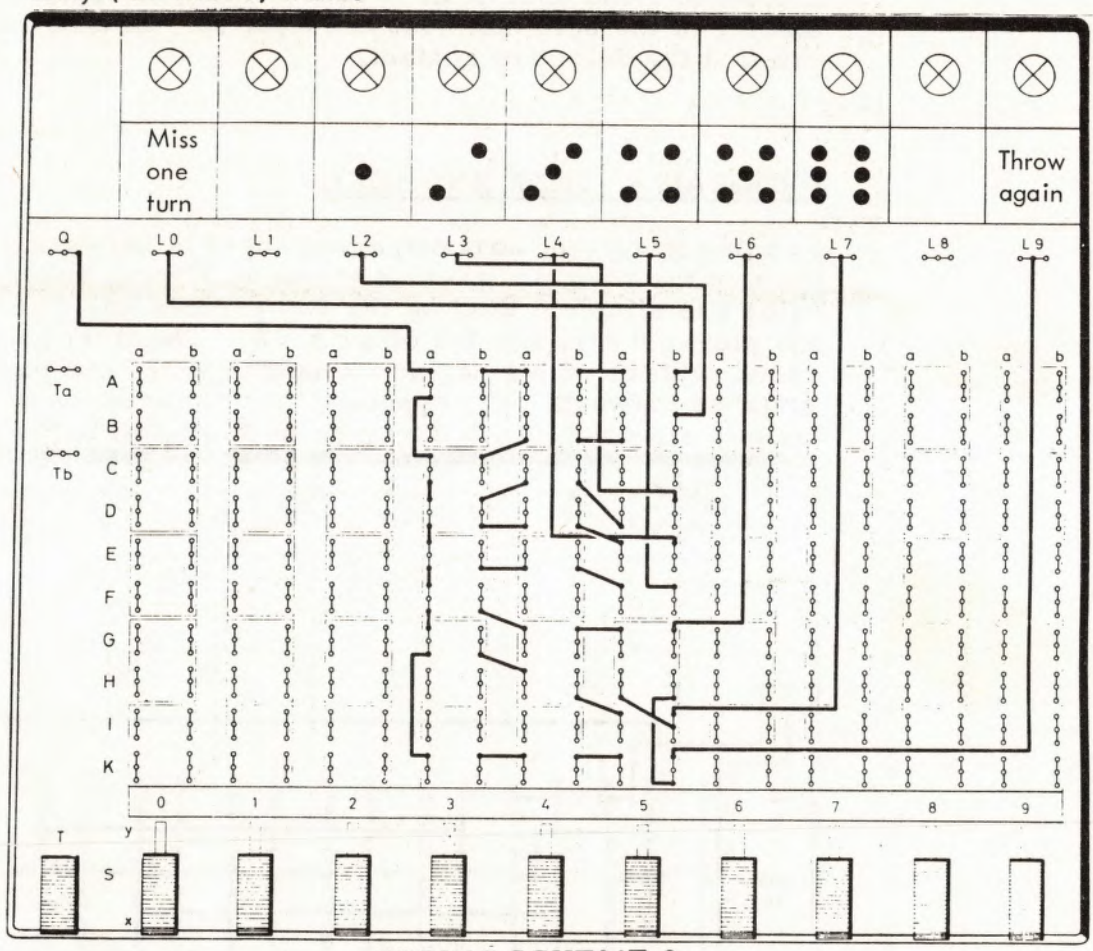
WIRING SCHEME 3b

To make life even more difficult for your partner, we have added a little surprise. When your partner thinks that he has ·ured out a "system" and can throw a six whenever he feels

like it, move Slider 9 (the panic button) to y. Suddenly your partner can no longer throw a six. By switching Slider 9 back and forth occasionally, you will keep him nicely confused.

Our third game of dice involves even more chance; if you play it quickly enough, it is almost impossible to figure out which slider will give what result.

Follow Wiring Scheme 3c carefully; this time we will be using only three Sliders: 3, 4 and 5. This is a game for two or three players. Each player, in turn, can move any one of the three Sliders either backwards or forwards. Make a note of the score of each "throw". The first player to reach fifty (or more) wins.



WIRING SCHEME 3c

REAL COMPUTERS WORK THIS WAY TOO!

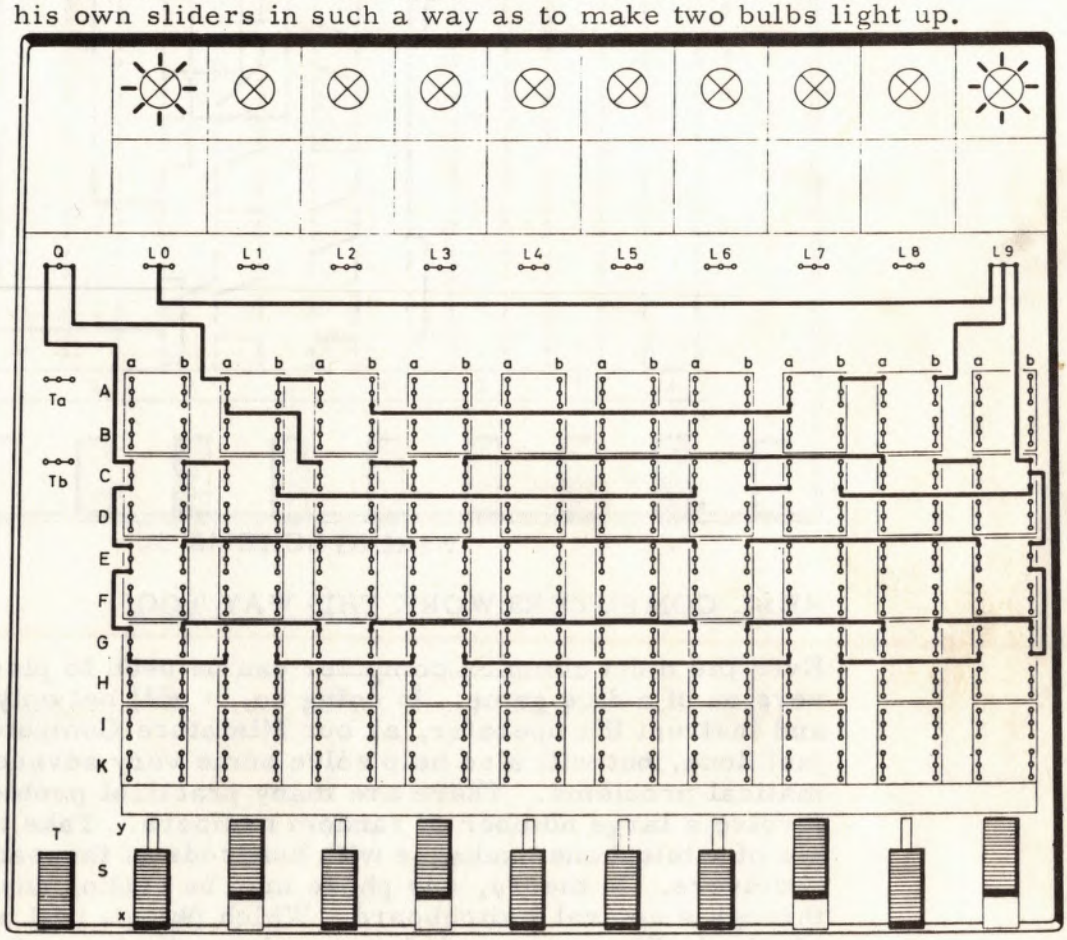
Even the most complex computer can be used to play its own version of a dice game. In doing so, it will not only amuse and instruct the operator, as our Miniature Computer has just done, but will also help solve some very advanced mathematical problems. There are many practical problems which involve a large number of random numbers. Take the example of a telephone exchange with hundreds of thousands of receivers. In theory, any phone may be calling any other through a central switchboard. Which phones will be calling which, is like a game of chance, where the laws of probabi-

lity rule. The computer can very rapidly copy such a situation, and thus help in the design and testing of large telephone networks. Many problems in mathematics, nuclear physics and other scientific areas involve the same type of approach; and to an unaided mathematician, the computer offers a means of solving the impossible.

Hint 2: It may happen that you do the wiring for a program correctly, but still do not get the required results. The reason for this may be because one of the contacts is not properly connected. If this happens, first check that the wiring is correct; then check that all the contacts are properly pushed in. If a bulb still refuses to light up, take a long wire and put one end of it in Q. Place it into the various switches leading to the bulb that does not light up. In this way, you can find the defective contact.

PROGRAM 4 - HIDE AND SEEK

For this program, no transparent sheet is necessary. Just wire up the program area as shown in Wiring Scheme 4. The rules are simple. Each of the two players has five sliders: one player 0 - 4, and the other 5 - 9. The first player covers up his sliders with a napkin or handkerchief and pushes two switches forward. His opponent now tries to move two of his own sliders in such a way as to make two bulbs light up.



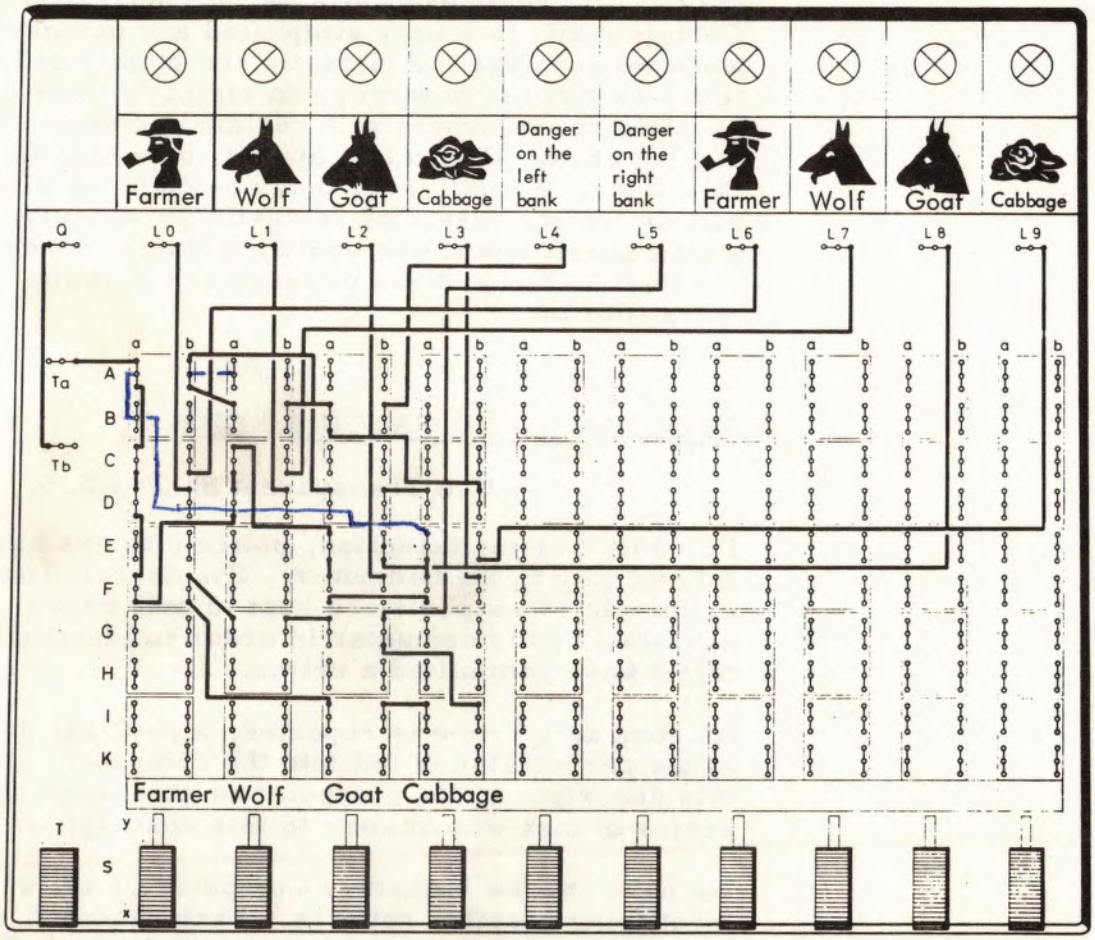
WIRING SCHEME 4

He is allowed three tries. If he gets it right the first time, he scores three points; second time - two points; and third time - one point: otherwise, no score. Then the players change places. All the sliders are pulled back to the x position. The second player takes the napkin, covers his sliders and pushes two forward. It is not easy to guess the other's slider positions. There are so many possible combinations!

PROGRAM 5 - CROSSING THE RIVER

- Use transparent sheet No. 5.

Here is a program based on an old riddle. It goes as follows: there was a farmer who had to cross a river in an old leaky boat. He had to carry across a wolf, a goat and a cabbage: why we don't know; the riddle does not explain. However, we are told that his boat was so old and leaky that it could carry only the farmer and one other object at a time.



WIRING SCHEME 5

What a problem! If the farmer goes across with his cabbage, the wolf will eat the goat. If he takes the wolf across, first, the goat will eat the cabbage. Thus, the only solution seems to be to take the goat over first, since the wolf will not eat the cabbage. But what should he take next? If he now takes the wolf across, the goat will get eaten while the farmer

returns to get the cabbage; however, if he takes across the cabbage and then comes later with the wolf, there won't be any cabbage left!

This whole rural drama is programmed for you on your Miniature Computer. To move an item across the river to the right bank, move the slider to y. To return to the left bank, move the slider back to x. Your computer will not solve the problem for you by itself, but it will help you solve it. You will be able to see immediately which objects are on each side of the river. It will warn you as soon as any two passengers that cannot get along with each other, find themselves on the same side of the river, so that you can quickly remove one of them before they have a chance to eat each other up.

REAL COMPUTERS WORK THIS WAY TOO!

In this program, your Miniature Computer has predicted the outcome of an event if a certain decision has been taken. Certainly this is a very simplified and obvious case. But when we consider the fantastic complexity of large computers, we can see how they can logically work out all the possible consequences of a certain decision. Thus, when playing chess, they might appear to be able to look into the future. Actually, all they are doing is carrying out simple logical reasoning fantastically quickly. A man with a pencil and paper would eventually reach the same result if he had endless patience and perhaps ten or twenty extra lifetimes!

PROGRAM 6 - TO CATCH A ROBBER!

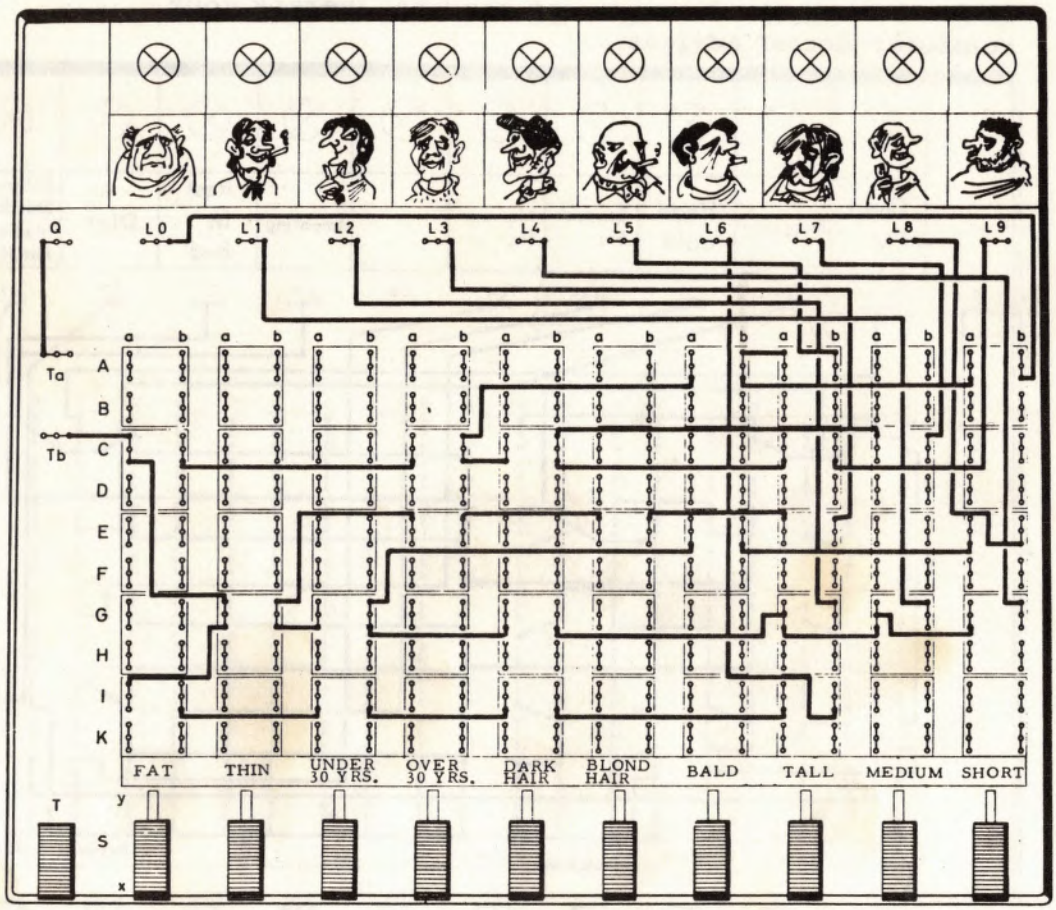
- Use Transparent Sheet No. 6.

In modern crime detection, police officers have a powerful new ally in the computer. The descriptions of known criminals or suspects are kept in data banks, which are searched by the computer in order to identify those that might have committed a crime.

As soon as a crime is reported, a physical description of the perpetrator is fed into the computer. Based on this description, the computer is able to quickly name a series of men who answer to this description.

We have set the Miniature Computer to assist you in identifying possible robbers. Each criminal answers to a unique set of physical characteristics. Your Miniature Computer can track down 10 criminals. Twenty-six physical descriptions also apply to honest citizens whose names are not found on the light panel.

Since we didn't happen to have on hand a real criminal, we had to imagine the characteristics of some thieves. If you wish, you can plan and program a sort of tracking system for your immediate familty and relatives. We



WIRING SCHEME 6

would then advise you, for your own safety, not to divulge the fact that you have composed a criminal poster of people you know for your Miniature Computer!

PROGRAM 7 - THE COMPUTER-DOCTOR

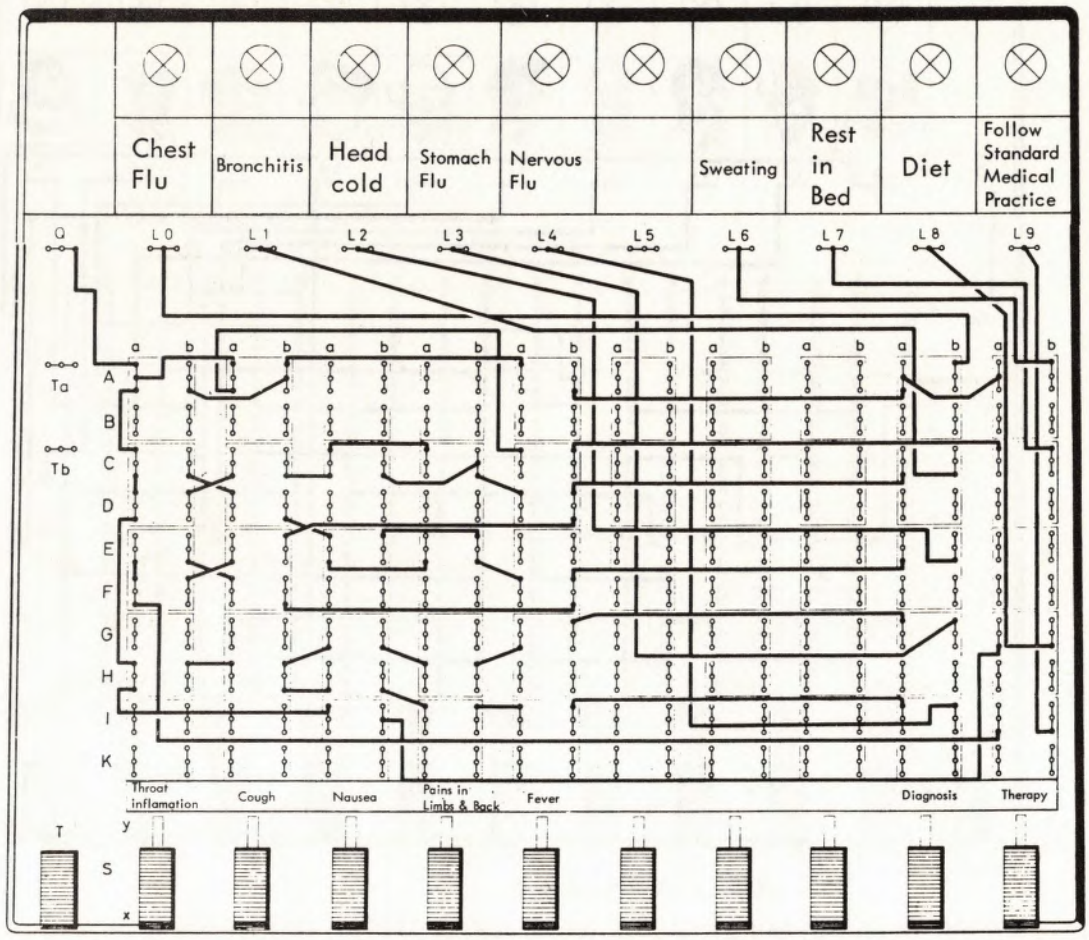
- Use transparent sheet No. 7.

Your Miniature Computer has now changed its profession. Imagine it in a white uniform with a stethoscope around its neck. With a new program, you can instantly change your Miniature Computer into a medical doctor.

Your Miniature Computer will patiently listen to you as you complain of all your aches and pains. It will give you a medical name for your illnesses and will also tell you what must be done to cure you.

Wiring Scheme 7 shows you the inside workings of this automatic doctor. After you have finished the wiring, you can show your complaints on Sliders 0 to 4 (the symptoms). Push Slider 8 (the diagnosis) forward, and you will immediately find out the name of the disease from which you suffer. To find out what you must do to get

better, return Slider 8 to its first position and push Slider 9 forward. Now you can see what therapy your computer-doctor advises.



WIRING SCHEME 7

REAL COMPUTERS WORK THIS WAY TOO!

Today there are large electronic computers that are programmed to give accurate medical diagnoses. They consider all the patient's symptons: temperature, pulse, heartbeat, breathing, his pains and other symptoms; and immediately indicate the suspected disease, and propose remedies and medications according to the most advanced medical literature. These large computers reach their conclusions basically in the same way as our Miniature Computer does, except that their branching possibilities are thousands of times more complicated and they are instantly able to use an entire medical library that has been fed into them.

In medicine, the practical value of the computer is that it can learn tens of thousands of combinations of symptomsand thousands of diagnoses. The computer is programmed according to the diagnoses of the best medical specialists. It is carefully tested, and any mistakes that are found are corrected. Obviously, these electronic machines can never compare to or replace a real doctor. What they actually do is to put a good doctor on

the right track when dealing with a very complicated set of symptoms. Instead of searching for hours, or maybe even days, the computer can lead him very rapidly towards the most likely answer.

LOGIC

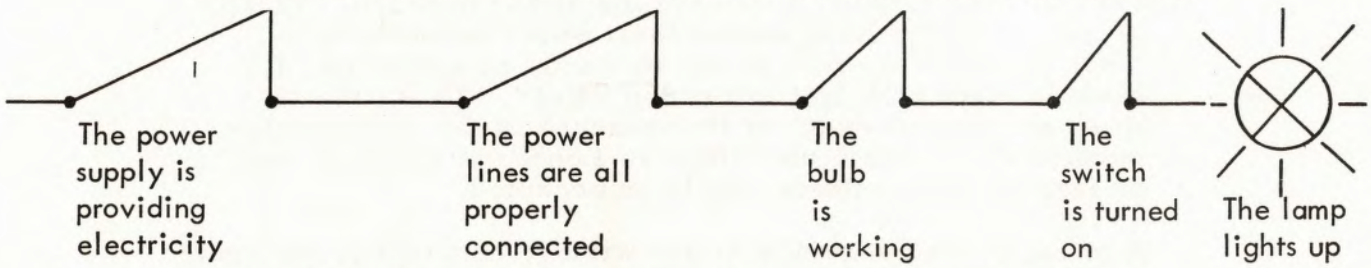
Up to now you have been getting to know your Miniture Computer. Now, we will actually investigate the basic laws followed by the computer in its operations - the Laws of Logic.

The two types of logical circuits we will be working with most frequently, are simply called the AND circuit and the OR circuit.

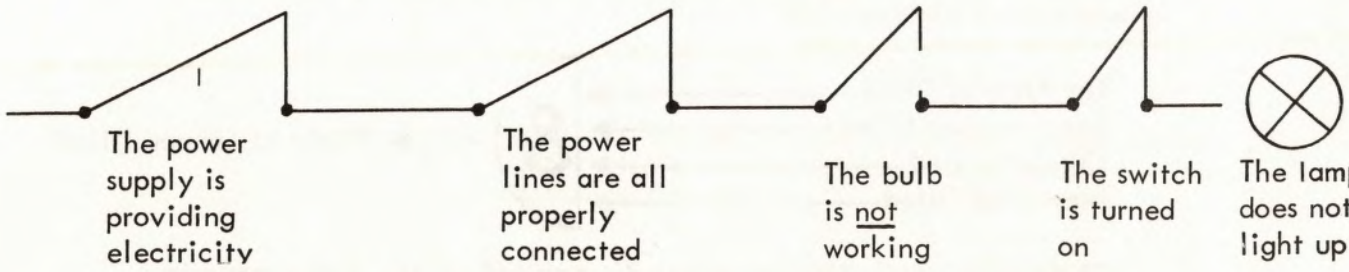
Let us start with the AND circuit. Tom and Tim, two twins with twin beds, who can't agree on anything, share a single bedside lamp. If there were only one switch to turn the lamp on, there would be trouble. If Tom had the switch, he might turn on the light when Tim wanted to sleep. If Tim had the switch, he might disturb Tom. How can we save the situation?

Obviously, a fair solution might be to create a circuit which would light the lamp only if both Tom AND Tim wanted it on. Such a circuit would involve two consecutive switches; one near Tom's bed, the other near Tim's. The light would turn on only if both Tom AND Tim turned their switches on, and for this reason, in terms of Logic, this is known as an AND circuit.

This Logical Scheme may be changed into an electrical scheme, **where each switch represents one of the conditions. as follows:**

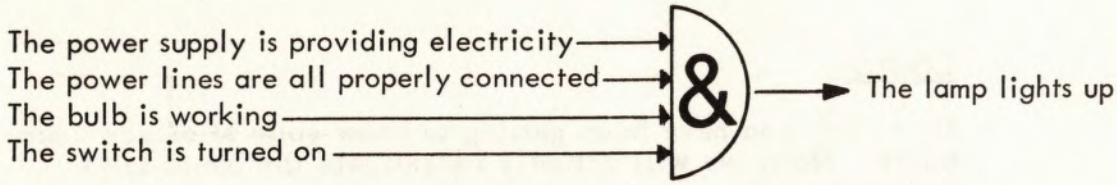

The power supply is providing electricity
The power lines are all properly connected
The bulb is working
The switch is turned on
The lamp lights up

Each switch thus represents one of the logical conditions. If any one of the switches is not closed, the lamp will not light up.


The power supply is providing electricity
The power lines are all properly connected
The bulb is not working
The switch is turned on
The lamp does not light up

In the same way as we have changed a logical scheme into a system of electrical connections, we can wire our Miniature Computer to solve logical problems electrically.

Looking at our AND circuit in terms of Logic; consider the following: WHEN electricity is being supplied AND the power lines are connected, AND the bulb is working, AND the switch is on, the lamp will light up. If any of these conditions is absent, the lamp will not light up. We can represent this in the following scheme:


The power supply is providing electricity
The power lines are all properly connected
The bulb is working
The switch is turned on
&
The lamp lights up

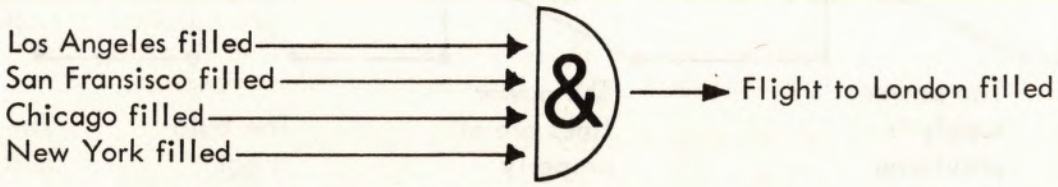
PROGRAM 8 - PASSENGER BOOKING WITH THE AND CIRCUIT

-Use transparent sheet No. 8.

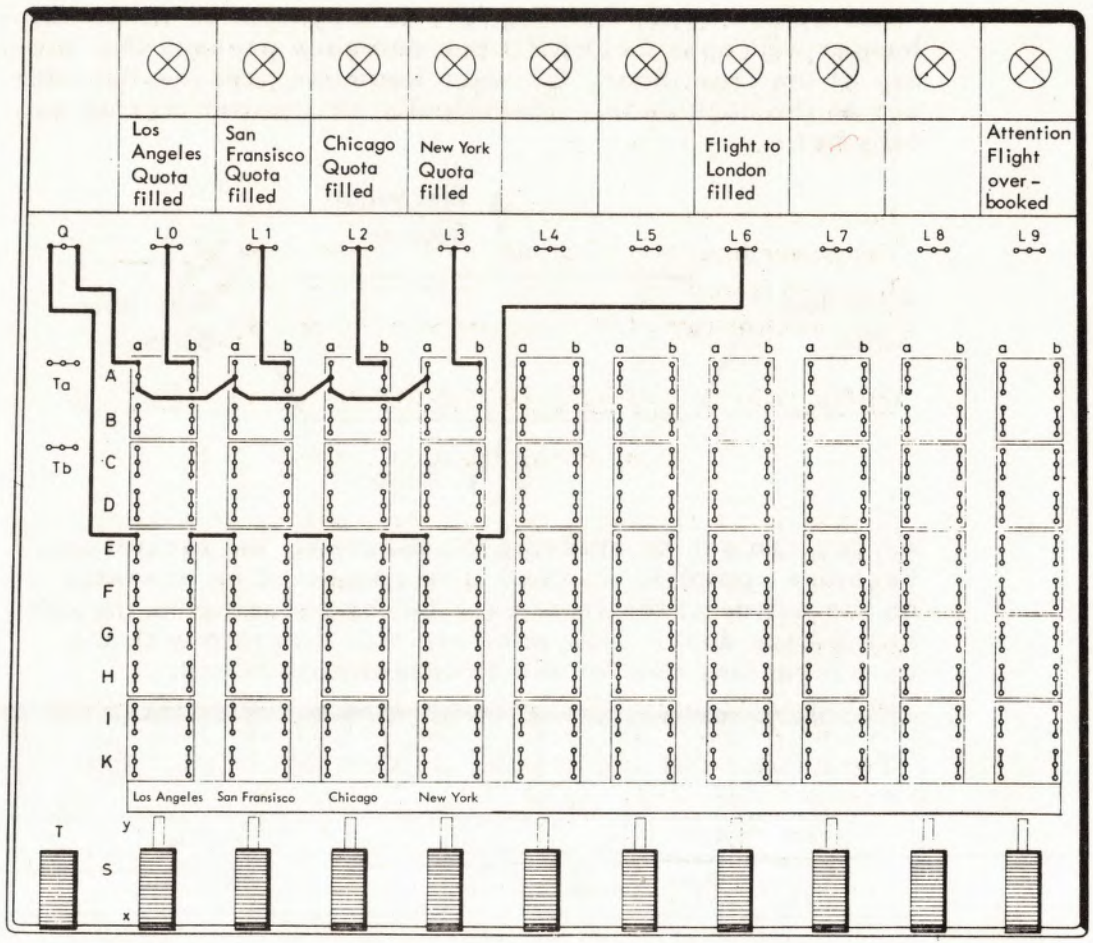
In the next few pages, we will be working on a simplified version of a program currently used by all major airlines. A central computer, located at the head office, communicates with each branch input at the various terminals served by the airline.

In our imaginary example, we have a flight that starts out in Los Angeles and goes to San Francisco, Chicago and New York. At each city it picks up passengers, and then finally crosses the ocean to London. The plane can carry a maximum of one hundred passengers. In order not to overload the plane, each airport is given space for twenty-five passengers. As soon as twenty-five passengers have reserved at each of the airports, a message is sent to the central computer "Passenger quota filled". On your Miniature Computer, you represent the quota of Los Angeles being filled by pushing Slider 0 forward in the direction of the arrow. As each of the other three airports fills its quota (in whatever order), you move the slider forward. Finally all four airports have booked their share of passengers. When you push the fifth slider forward, an extra light flashes on stating "Flight to London filled". Thus, the airline can now warn each of its branches in the various airports that this particular flight to London is filled up and no further reservations should be accepted.

Now let us look carefully at our wiring. Logically, we could say that WHEN Los Angeles AND San Francisco AND Chicago AND New York have sold out their share of tickets, THEN the flight to London is filled. In a logical scheme, we would show it as follows:



The same type of scheme can be applied to the Programming Board on your computer (Wiring Scheme 8). Follow the wiring that leads from Q to E and then across Switches 0 - 3, finally connecting with Light 6.

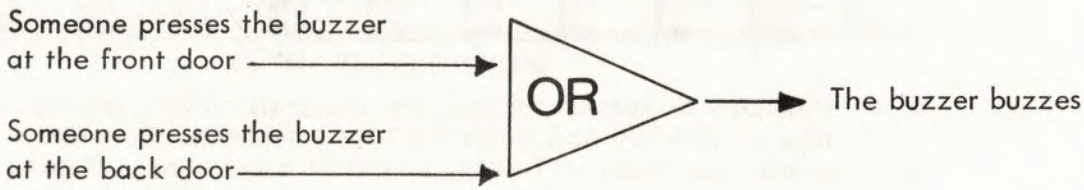


WIRING SCHEME 8

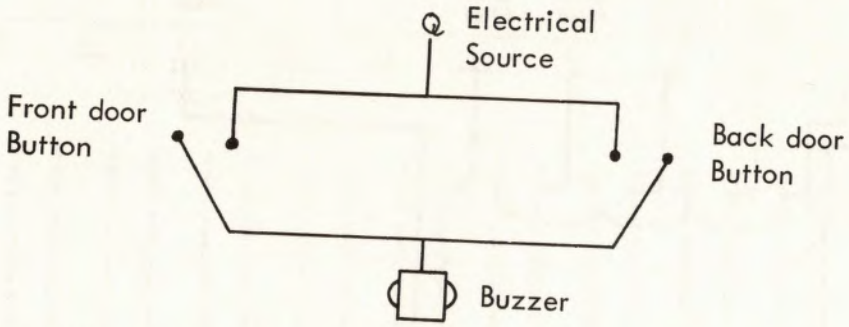
PROGRAM 9 - PASSENGER BOOKING WITH THE "OR CIRCUIT"

- Use transparent sheet No. 9.

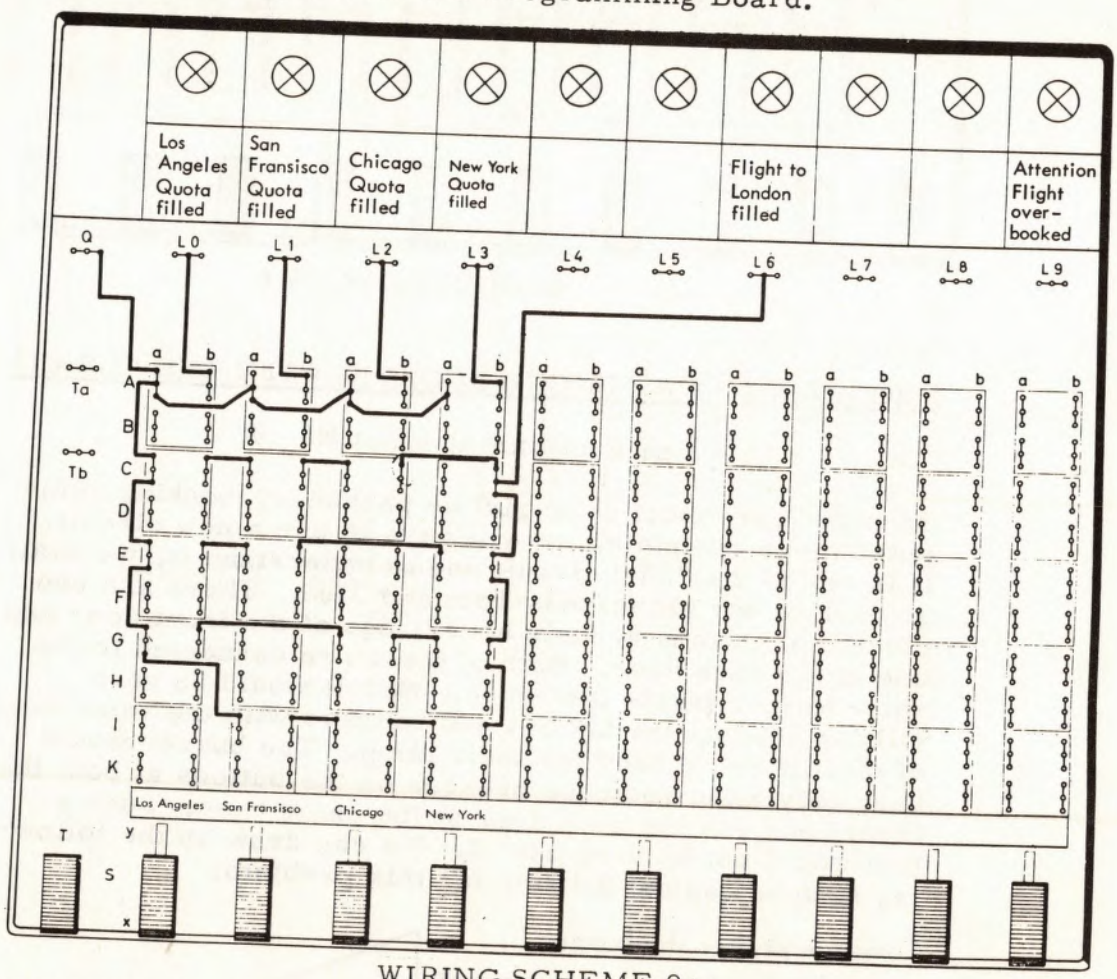
To make our model of an airline passenger-booking computer more complete, we will have to use a new circuit. It is called the "OR" circuit and to understand it, we shall go back to our old friends Tom and Tim. There are two electrical buttons at their house; one at the front door and one at the back door. Both of these are connected to the same buzzer inside the house. Now it would be very silly to connect the buttons and buzzers with the same sort of circuit as we used for their lamp. The buzzer would buzz only when someone presses on the buttons at both the front door and the back door at the same time. Such a bell would not be very useful. We can draw up the following simple Logical Scheme for this problem:



What we are really looking for, is a system by which the buzzer will buzz EITHER when someone presses the buzzer at the front door, OR when someone presses the buzzer at the back door. Electrically we can set this up as follows:



As you can see by studying the diagram, the circuit now becomes complete whether it is connected by pressing on the button at the front door OR by pressing on the one at the back door. Now compare this diagram with the corresponding one for the Programming Board.

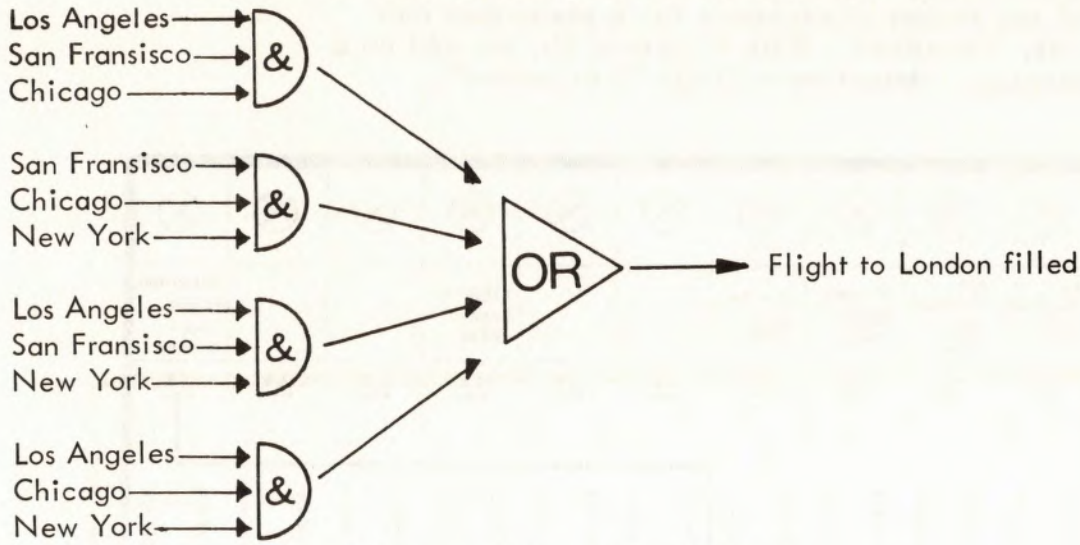


WIRING SCHEME 9a

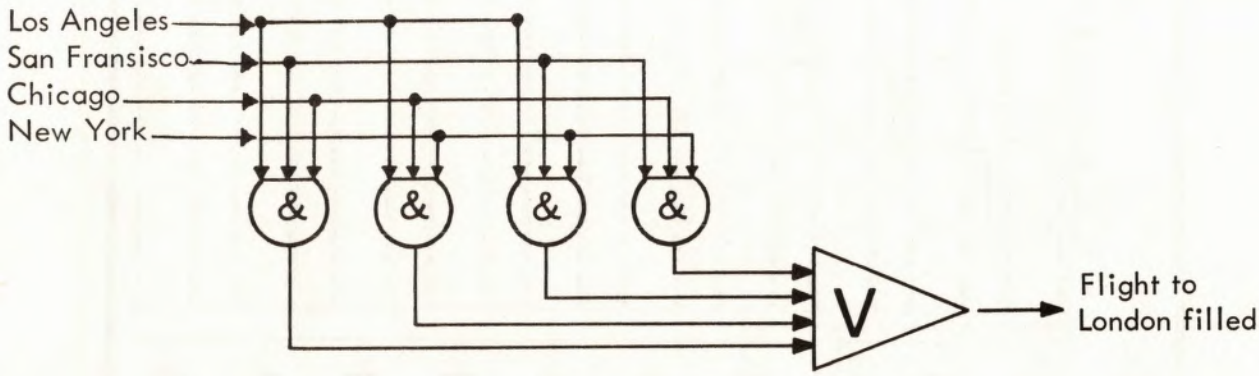
In this Wiring Scheme, we will be using the "OR" circuit in a very simplified and unlikely way. Suppose that the airline will use this particular flight to carry passengers from the first airport that reports all one hundred seats

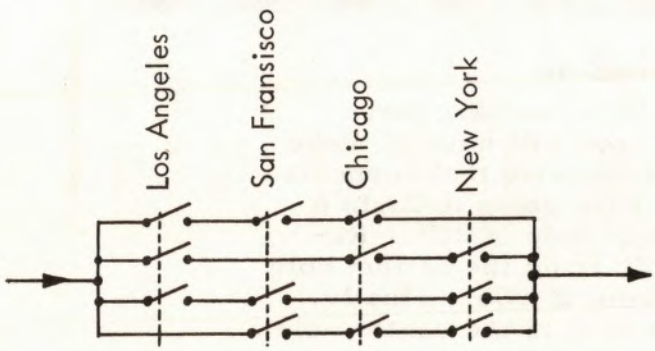
filled. (There will be more planes for the other airports). When you complete the wiring, notice that as soon as any of the airports reports a full load, the signal lights up for "Flight to London filled".

A much more complicated program is based on the following assumptions: each airport is given a quota of thirty seats to fill. We assume that when three of them report their quotas filled (3 x 30 = 90), the fourth airport has at least ten passengers. We can represent the Logical Scheme of this problem as follows:

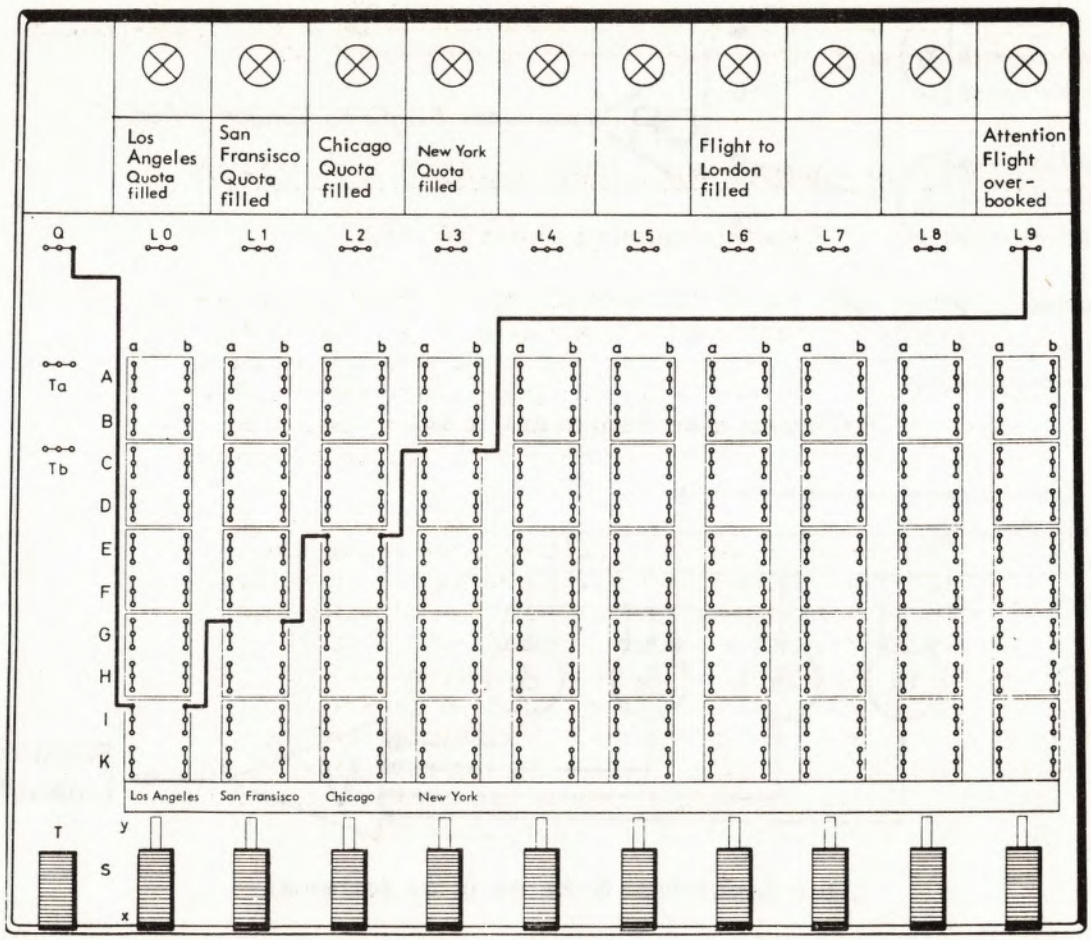


We can also represent it as:



Our Electrical Scheme is as follows:

Carefully compare this Electrical Scheme with Scheme 9a. If you leave out the wiring along row A (which lights up the various airport indicators), you will notice that the horizontal rows of AND circuits are set up just as in the Electrical Scheme. This is a permanent wiring characteristic of these switches: the AND circuit being horizontal, the OR circuit vertical.

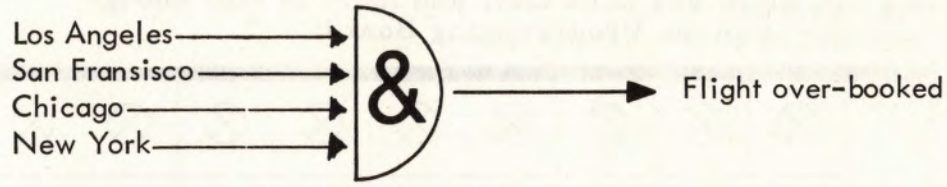
Now we can add another improvement to this program. Let us suppose that on a day of very heavy bookings, all four airports report that all their places are filled at the same time. This would result in a load of a hundred and twenty passengers for a plane that can carry only a hundred. With Program 9b, we add on a light warning: "Attention - Flight Overbooked".



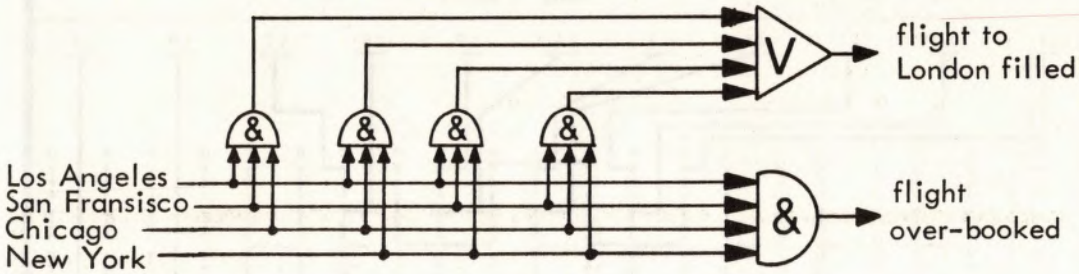
WIRING SCHEME 9b

Program 9b fits right on the 9a which you have just completed. To do this, however, you will have to make a few adjustments. First remove the wire that connects 2Cb with 3Cb. Next, remove the wire going to Light 6 from 3Cb, and place it in the middle hole of 2Cb. Remove the wire that comes from 3Eb from the bottom hole of 3Cb and place it in the bottom hole of 2Cb. Finally, move the wire from the right hole of Q to the center hole.

In the Logical Scheme, our addition to the program is:



Therefore the Logical Scheme of the final program is:



PROGRAM 10 - WHAT WILL THE WEATHER BE LIKE?

- Use transparent sheet No. 10.

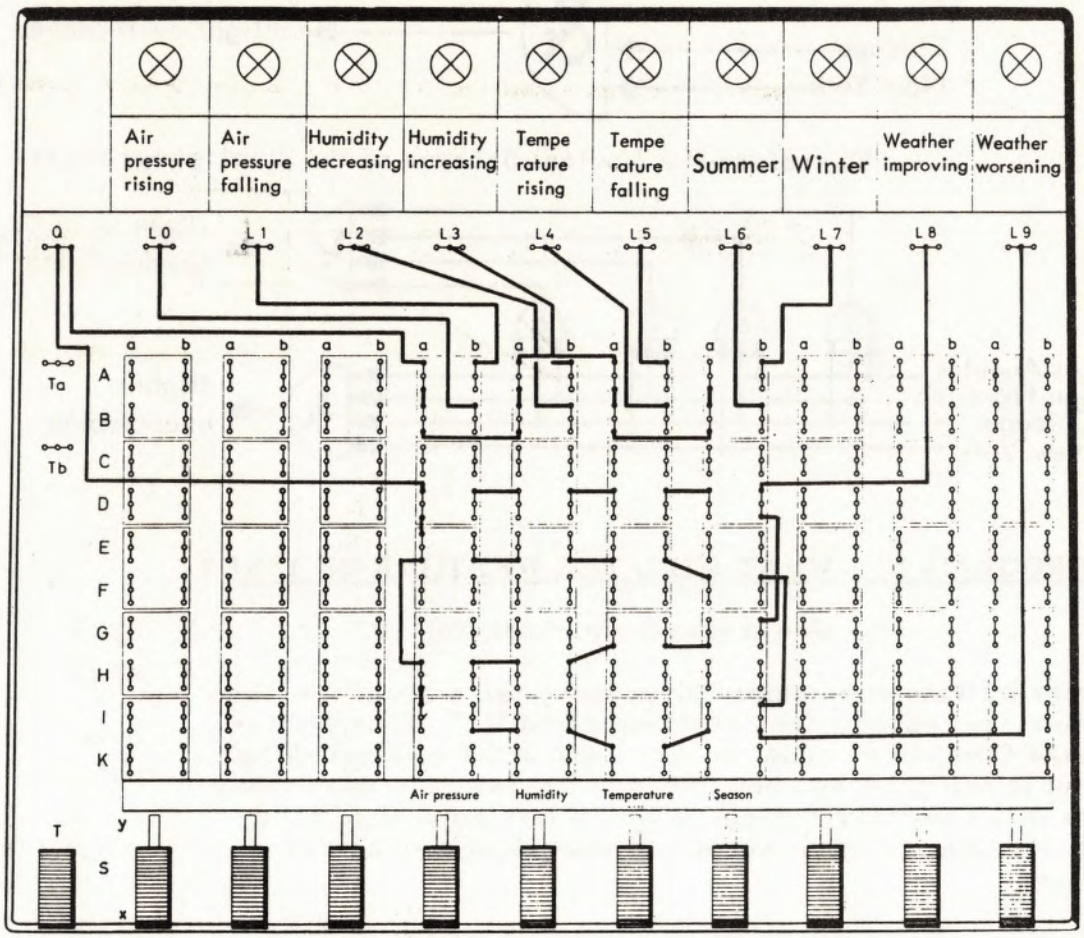
Mark Twain once said: "Everybody talks about the weather, but nobody does anything about it." Your Miniature Computer cannot do much about the weather either, but it can show you how large computers are used today to make weather forecasts with a fair amount of accuracy. Let us try it out in its new job of weather prediction.

In this scheme for weather prediction, we make use of four different factors, or variables as the scientist would call them: season, air pressure, humidity and air temperature. You can easily find out the facts you need for the Input. Season is certainly no problem. If you have a barometer, you can determine the fall or rise of air pressure. A common thermometer determines temperature. As for humidity, if you do not own a hygrometer, listen to the radio or trust yourself to guess the humidity approximately.

Of course, the more variables we put into our program, the more likely our prediction is to be accurate. Professional meteorologists work with many other factors which are gathered from precise and expensive equipment scattered all over the world and which the average person cannot obtain.

With only four variables, no one would expect us to be able to accurately predict the weather in advance. But although our program is far from complete, it is still basically correct. Obviously, this program is only a simplified example. If you have a special interest in the weather and learn more about it, you will be able to modify this program and add more variables.

The knowledge of programming you have gained up to now will allow you to do this, and there is still enough room for it on the Programming Board.



WIRING SCHEME 10

REAL COMPUTERS WORK THIS WAY TOO!

This program actually has some similarity - although in a very simplified form - with the way real electronic computers are programmed to accurately predict the weather. Naturally, the large computers can use many hundreds of variables and have accurate weather data gathered from all parts of the world available to them. The large computers evaluate this data, compare it with previous weather situations and come up with a surprisingly accurate picture of future weather conditions.

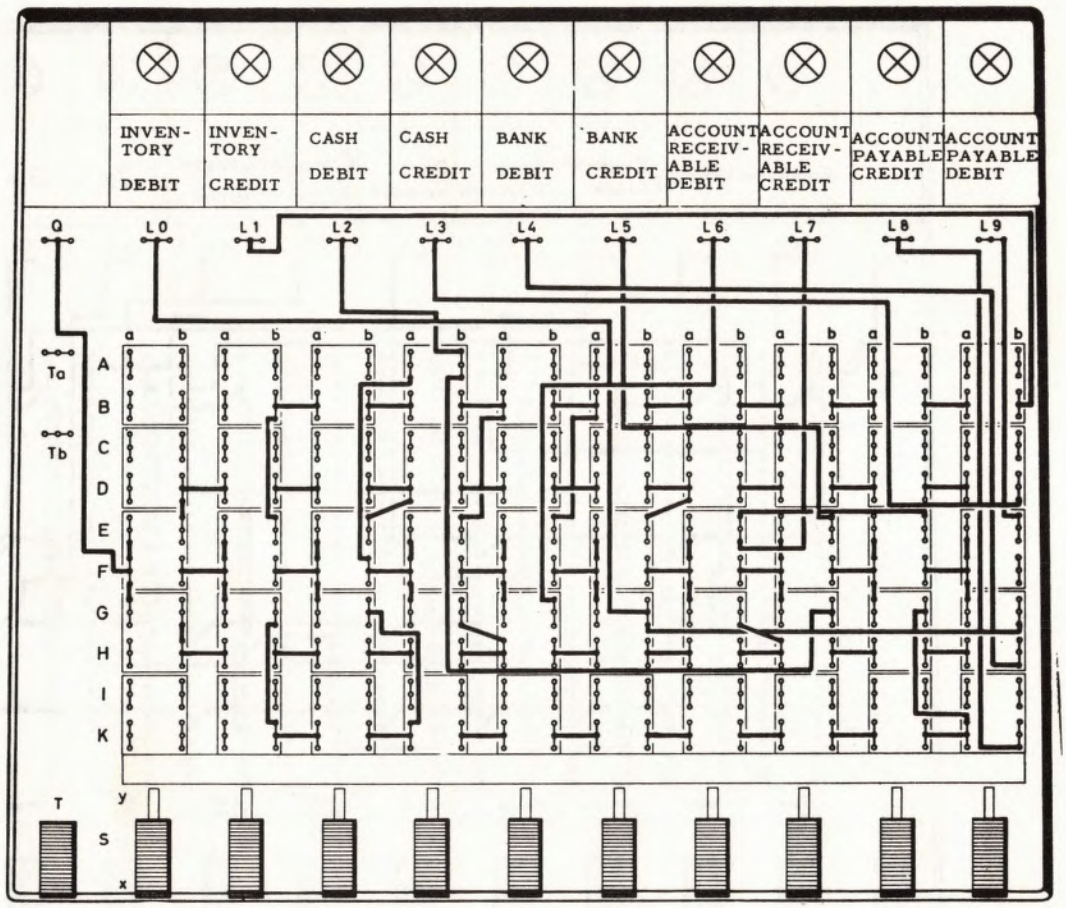
Weather predictions of this sort are only possible because of the development of the electronic computer. Before computers, weather predictions had to be done by comparing large tables of figures manually. The number of variables needed is so great that by using only an ordinary office calculating machine, the weather would be over long before the prediction has been completed.

PROGRAM 11 - BOOKKEEPING

- Use transparent sheet No. 11.

This program transforms the Miniature Computer into a bookkeeping reference guide. It is true that it doesn't register miscellaneous sums, but it does indicate the necessary bookkeeping entries required in a business transaction.

Since the slider description spaces are very limited; we list below the transactions which correspond to the respective sliders.



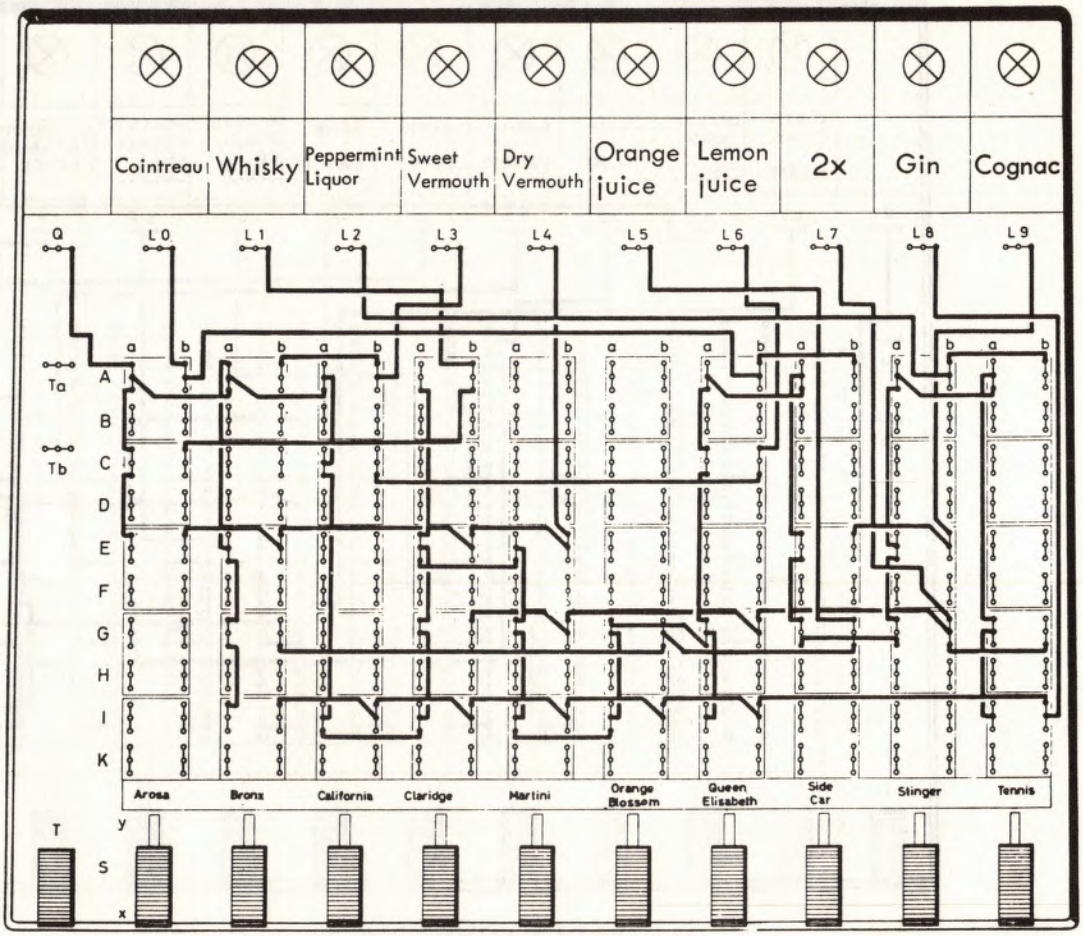
WIRING SCHEME 11

SLIDER	KEY WORDS	DESCRIPTION
0	Bank Deposit	We bring money to the bank
1	Goods Return	We send goods back to the supplier
2	Cash Payment	We pay supplier
3	Cash Sale	We sell goods for cash or check
4	Term Sale	We sell goods on credit
5	Cash Purchase	We buy goods for cash
6	Customer Payment	A customer pays by check
7	Bank withdrawal	We withdraw money from bank
8	Order Transfer	We transfer an order to a supplier
9	Term Buying	We buy goods on credit

PROGRAM 12 - THE COMPUTER-BARMAN

-Use transparent sheet No. 12.

Here is a program that would be equally useful, on a large scale, in a chemical factory and on a small scale, in a bar. A computer-barman is never too tired to remember the ingredients of a favorite cocktail. You can easily see how this program could be adopted under different circumstances to help automatic production in a pharmaceutical or other type of factory. Instead of turning on a light, as in our Miniature Computer, the wiring would be connected to switches that control the components of the assembly line.



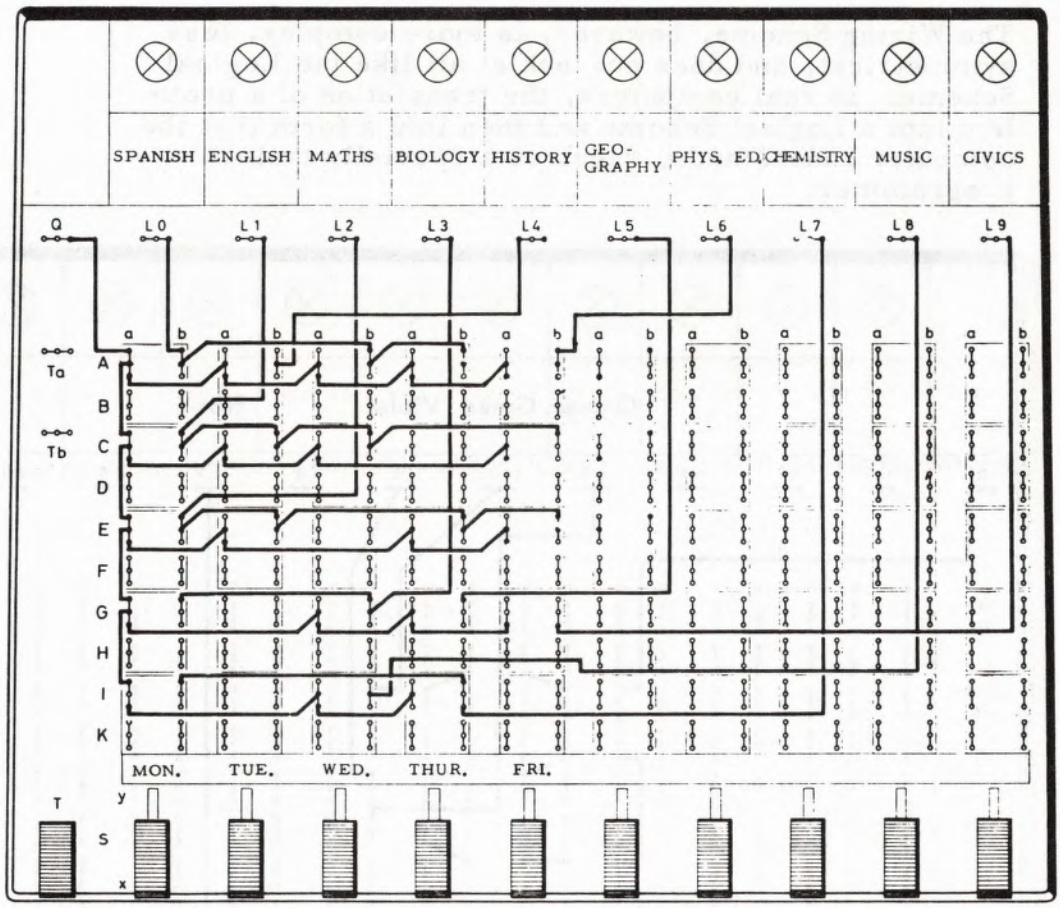
WIRING SCHEME 12

PROGRAM 13 - STUDENT TIME-TABLE

-Use transparent sheet No. 13.

Here we have a really practical program which will show you how to set up a time-table or school course schedule on your Miniature Computer.

Of course, everyone should set up his program according to his own schedule. The Wiring Scheme is only one example.



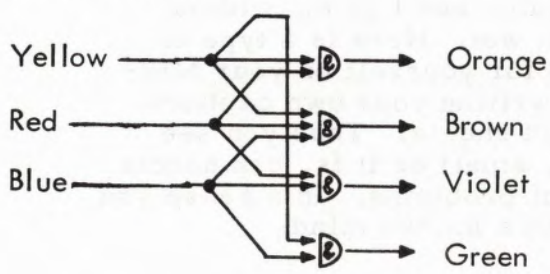
WIRING SCHEME 13

A similar schedule could be set up by the physical education instructor, to show the schedule for use of the gymnasium; or the chemistry instructor could schedule the use of the laboratory. Can you think of other applications of this idea?

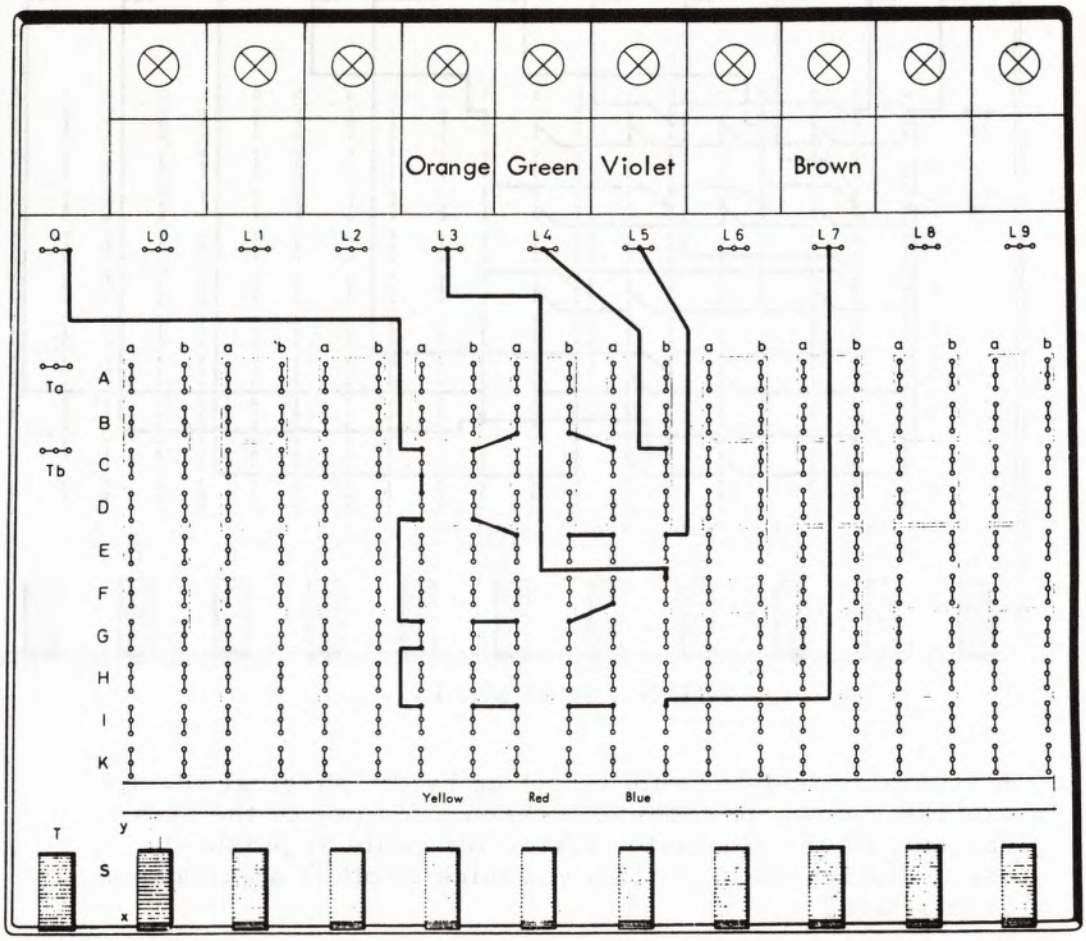
PROGRAM 14 - FROM COLORS TO LOGIC

- Use transparent sheet No. 14.

This is a very easy program. The problem of mixing primary colors to produce secondary ones is a simple one; however, this program does teach us something about computers. The Logical Scheme for this problem is as follows:



The Wiring Scheme, however, is more complex, less symmetrical, and does not look at all like the Logical Scheme. In real computers, the translation of a problem into a Logical Scheme and then into a form that the computer understands, requires a specially trained programmer.



WIRING SCHEME 14

We see that our Miniature Computer now works according to the Laws of Logic. It can reach the conclusion that if you mix yellow and red, you come up with orange. To put this in a logical form: IF there is red AND it is mixed with yellow, THEN you will get orange.

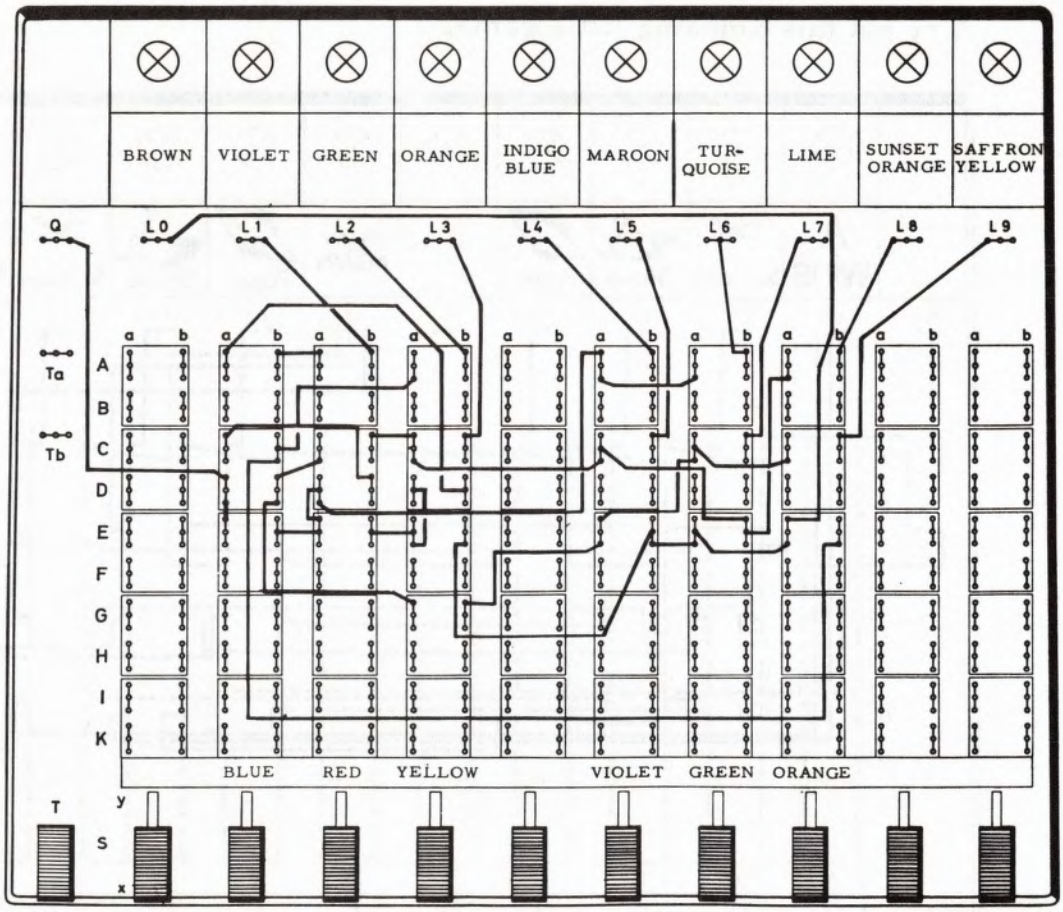
This type of reasoning is not limited to mixing colors, but can also be applied to a wide variety of logical statements. For instance, IF it rains and I go out without my umbrella, THEN I will get wet. Here is a type of logic that you can easily wire for yourself on your Miniature Computer, merely by writing your own captions on one of the blank transparent sheets. Thus you see that your Miniture Computer, small as it is, can handle many different kinds of logical problems. In a sense you can say that it thinks, just like a human mind.

PROGRAM 15 - PRIMARY, SECONDARY AND TERTIARY COLORS

By Marcel Tremblay,
Lac St. Jean, Quebec.

- Use Transparent sheet No. 15.

In the previous experiment we saw how primary colors (e.g. yellow, red, blue) could be combined to produce secondary colors (e.g. orange, green). In the same way, primary and secondary colors might be combined to give tertiary colors such as turquoise, maroon and lime.



WIRING SCHEME 15

Note that every time all the primary colors are found in a combination; brown appears.

Example: 1. Orange + Blue = Brown

Red + Yellow + Blue = Brown

2. Violet + Yellow = Brown

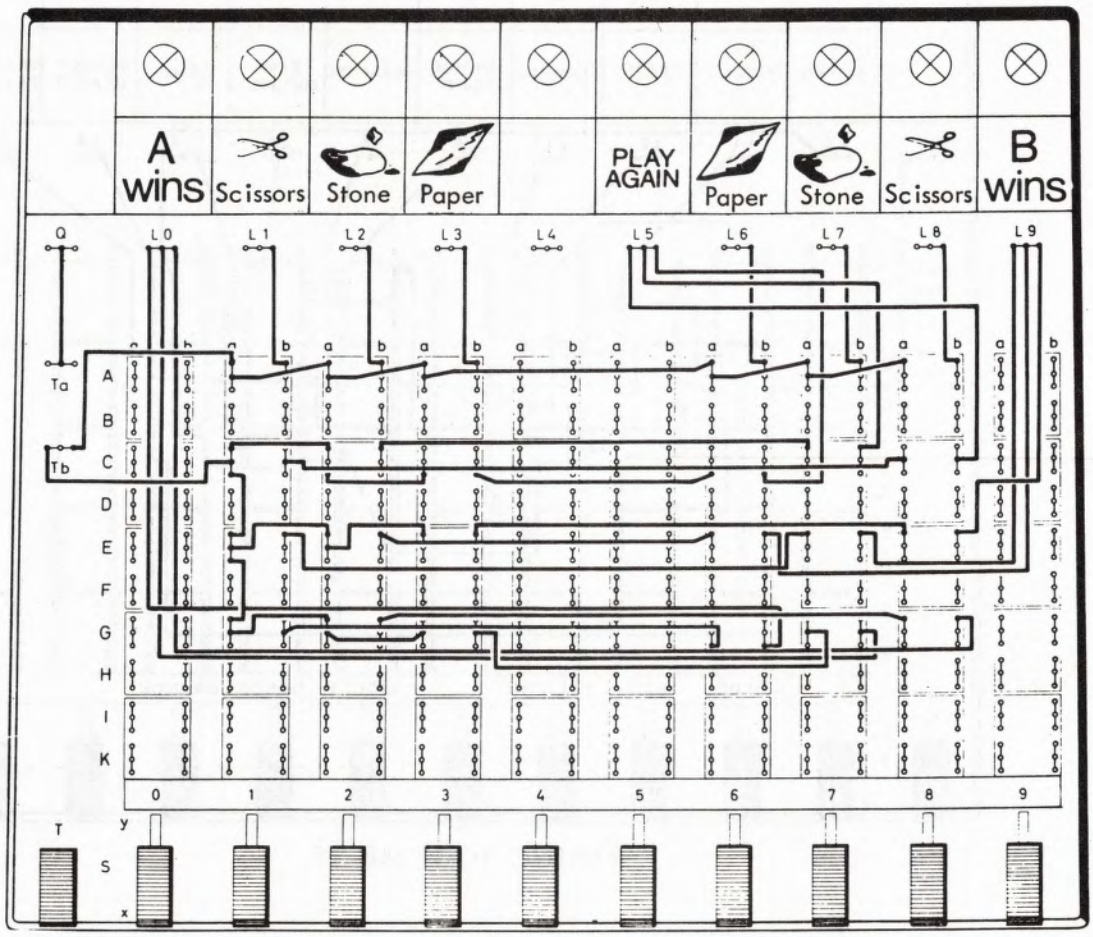
Red + Blue + Yellow = Brown

PROGRAM 16 - PAPER, STONE, SCISSORS

- Use transparent sheet No. 16.

This program is a game of chance for 2 players. The first player uses Sliders 1, 2 and 3, while the second uses Sliders 6, 7 and 8. The players hide their sliders. Each then moves one of his 3 sliders from x to y. Then they press down on the push button (T) to see who is the winner, if any. The game is scored as follows: the scissors beat the paper because they can cut it. The paper beats the stone because it can cover it. The stone beats the scissors because it can break them.

Try out this amusing little game.



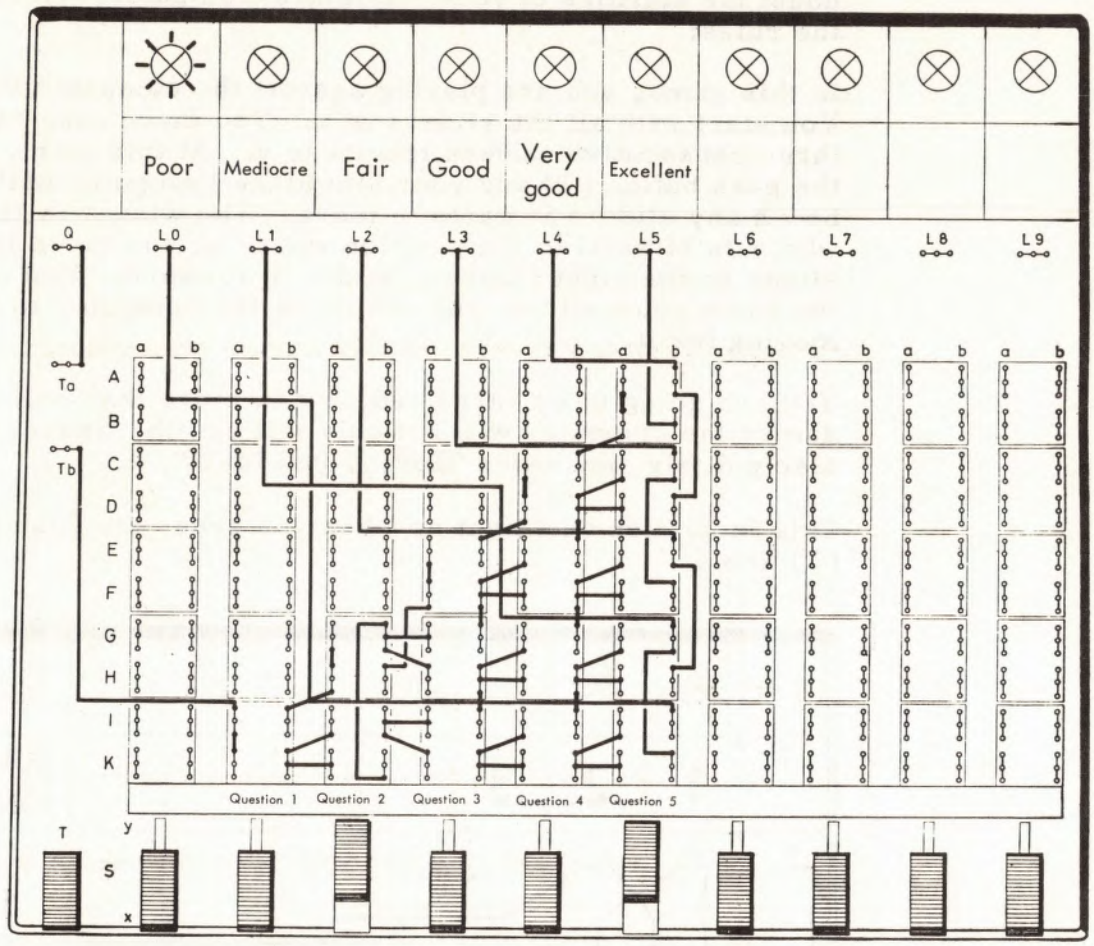
WIRING SCHEME 16

PROGRAM 17 - A KNOWLEDGE TEST

- Use transparent sheet No. 17.

Here is a knowledge test that you can give your relatives and friends. This program is based on your Miniature Computer's ability to count.

Here is how you do it. Prepare five fairly difficult questions in such a way that each is given 2 answers (we give one suggested set below). Each of the answers is called x or y. After you have programmed the test, you score it by setting the sliders according to the answers that are given.



WIRING SCHEME 17

SUGGESTED QUESTIONS:	x	y
1. - When did Columbus discover America?	1066	1492
2. - How far is the moon?	240,000 mi.	800,000 mi.
3. - What is the population of the United States?	450,000,000	200,000,000
4. - Who invented the telephone?	Morse	Bell
5. - What is the capital of Argentina?	Buenos Aires	La Paz

PROGRAM 18 - SORRY, YOU LOST!

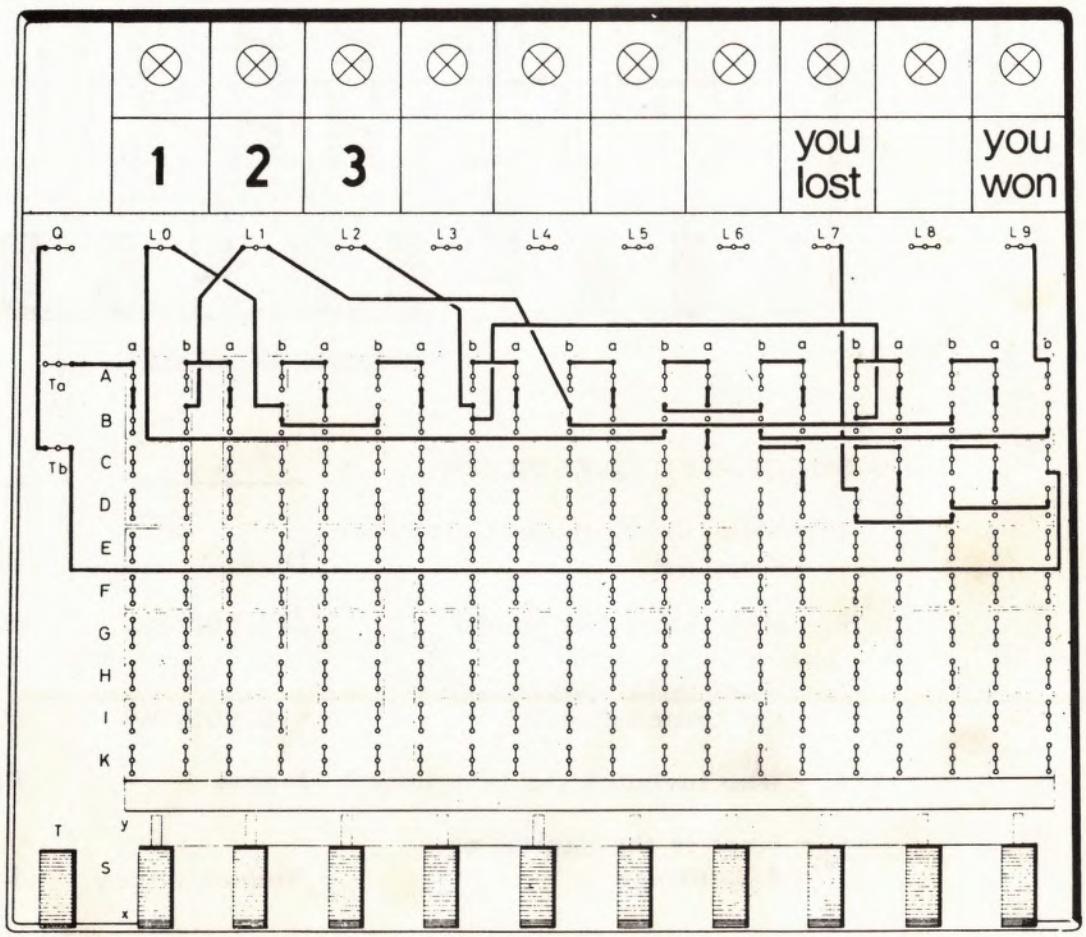
-Use transparent sheet No. 18.

Here is another game, one that you should show people who doubt the abilities of your Miniature Computer. Here are the rules:

In this game, you are playing against the computer itself. You start with all the sliders at x. You move one, two or three consecutive sliders from x to y. At this point, you press the push button (T) and your Miniature Computer will tell you how many sliders it wants to move. The winner is the player who puts himself in the position where he can move the last slider on the right (that is, Slider 9) forward. You may start the game yourself, or you can allow the computer to begin by moving for it.

You are going to be surprised, but in more than half of the games the computer will not only tell you the figure, but will also politely announce: "Sorry, you lost!"

This indication will light up when you press the push button (T) down.

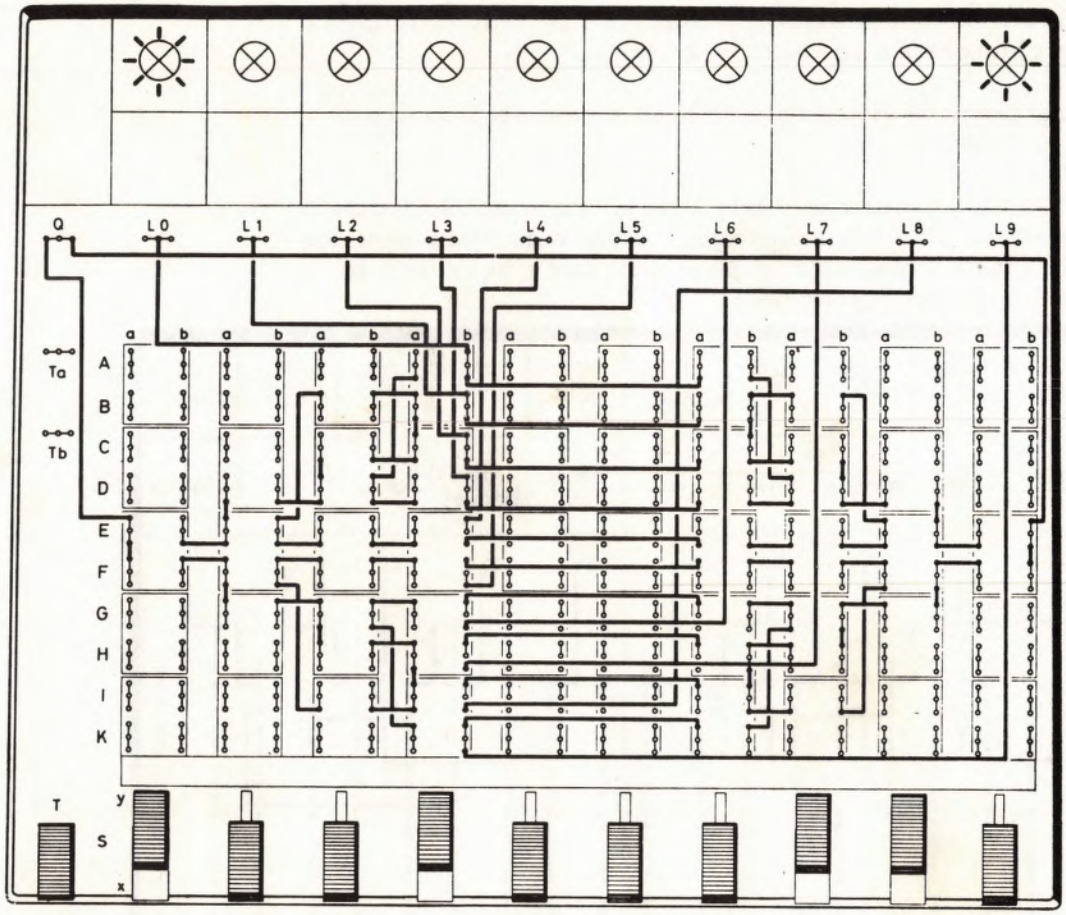


WIRING SCHEME 18

When you win because you are the one to move the last slider (Slider 9) from x to y, the Miniature Computer will be honest enough to recognize that you are the winner, and will tell you so. An indicator will light up saying: "Congratulations. You win!" Anyone who doubts the intelligence of your Miniature Computer will quickly change his mind after a few attempts at this game.

PROGRAM 19 - CAT AND MOUSE

You need no transparent sheet for this program. Just wire up the program area as shown in the Wiring Scheme. This is a game for two players; one player has Sliders 0 - 3, the other player has 6 - 9. Hence, each player has four. Before the play begins, Sliders 0, 3, 7 and 8 are moved forward.



WIRING SCHEME 19

Now a light will come on in the left-hand bulb, that is the mouse; and from the right-hand bulb you see the cat peering out. The player on the left is in charge of the mouse, and the right-hand player has the cat. The right-hand player must try to catch the mouse with the cat as quickly as he can. This happens when suddenly there is only one bulb alight. This means the cat has swallowed the mouse. The cat player will move one of his switches

as he pleases and the cat will jump into another bulb. Next it is the turn of the mouse player with the move of the switch; then the cat player, and so on. This game should be played quite fast. The cat player and the mouse player change over every few minutes, and the time for catching the mouse should be measured. The winner is the player who catches the mouse in the shortest time.

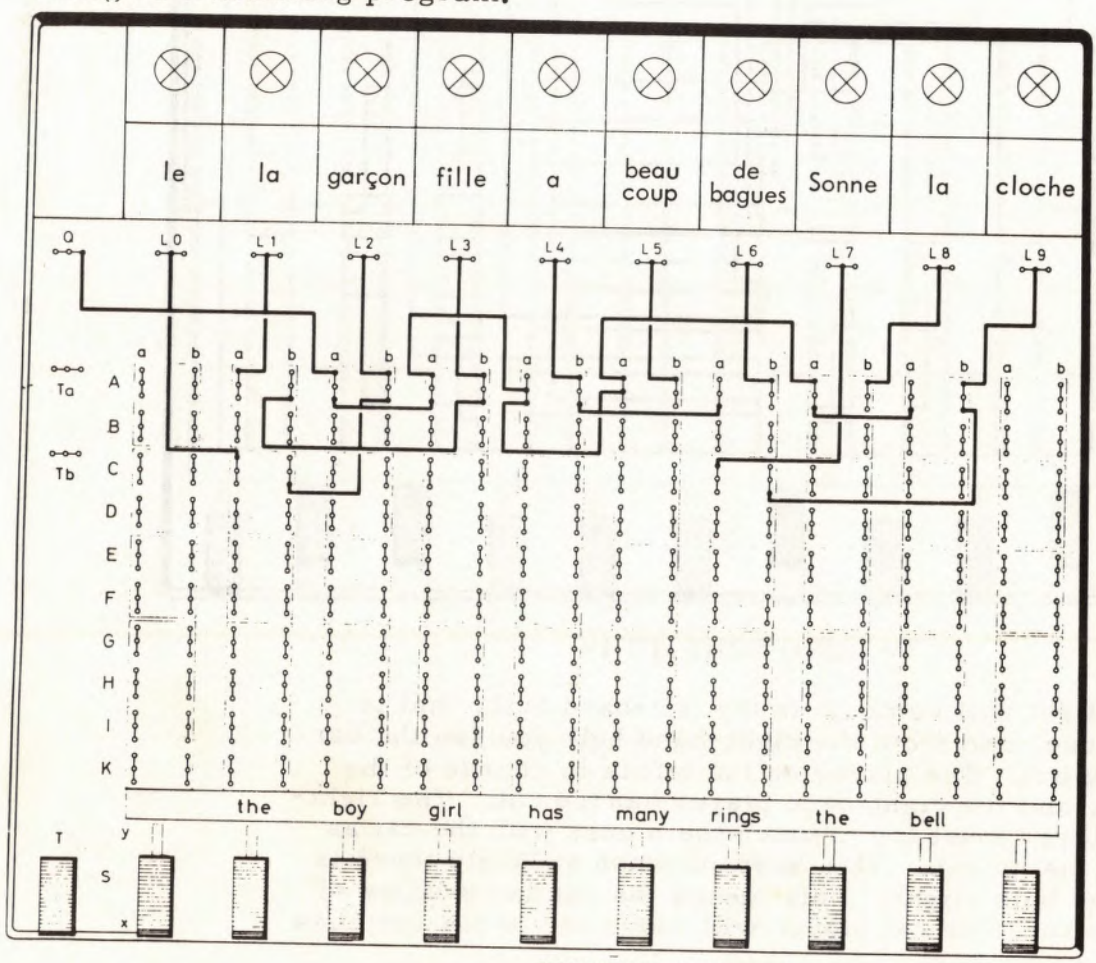
PROGRAM 20 - A TRANSLATING MACHINE

- Use transparent sheet No. 20.

If every language had the same grammar rules, it would be an easy thing to make a translating computer. Then it would be only a matter of matching the word in one language to the corresponding word in another language. The problem of translation by machine is that languages are very changeable in their grammar.

This program will illustrate this in a translation from English to French.

To show you some of the difficulties, we must tell you a little about the French language. This will show you the problems that a computer expert will face in trying to design a translating program.



WIRING SCHEME 20

Take the word "the" in English. In French it may be translated by either of two words: "le" or "la", depending on whether the word it goes with is masculine or feminine. "Garçon" is masculine, so when you want to say "the boy", you say "le garçon". "Fille" in French - girl - is feminine and so if you want to say "the girl", you must say "la fille". In order to translate properly from English to French, the translating machine must know this distinction. It must be able to choose "le" or "la" just as someone learning the language must do. You will notice in this program that our Miniature Computer actually does make this choice without error.

Here is another problem. Sometimes where we use the same word for two different things in English, another language may use two words. Take, for example, the word "rings". In English, we would use exactly the same word for the sentence "The girl has many rings", in which the word "rings" is a noun. In French, however, the two uses of "rings" are distinguished by being two different words. In the first case where "rings" refers to action, the French uses the verb "sonner". In the second case where "rings" refers to an object, something you wear on your finger, the French would use the word "bague". In order to translate efficiently from English to French, your Miniature Computer must be able to distinguish between the two uses of the word "ring". In fact, it actually can do this. If you experiment with this program and set up the two different sentences, you will see that your computer will choose the right form of the word "rings" to translate the two uses of the English word.

REAL COMPUTERS WORK THIS WAY TOO!

Although computers can match a word in one language to that of another and can even follow grammatical rules, they can never achieve a really natural translation from one language to another. For instance, one machine when translating a Biblical text from Greek into English translated "The spirit is strong but the flesh is weak", as "The whiskey is well-recommended but the meat isn't very good". This anecdote is not meant to convince you that our computer's translations are this bad. However, very strange interpretations of the foreign language often emerge; these might be acceptable for translating a scientific text but hardly for translating a novel or a poem.

PROGRAM 21 - THE COMPUTER-DETECTIVE

- Use transparent sheet No. 21.

We can now have our Miniature Computer play detective. Following certain clues which we will feed into it, it will solve a case with the skill of Sherlock Holmes.

Here is the problem: Mr. Smith, Mr. Jones and Mr. Davis are three residents of a small town, who like to meet frequently in a local restaurant. One of the three is the mayor of the town, the other is a druggist, and the third is the doctor. The question is: "What is the profession of each of these three men?"

Before you start to figure this out, we will give you three clues:

1. Mr. Smith and the doctor often like to tell little stories to each other.

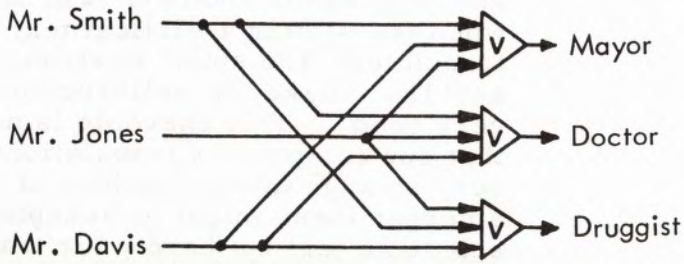
2. Mr. Jones and the druggist often play chess with each other.

3. The doctor and Mr. Jones frequently like to argue with each other about various questions.

To work out this problem on your Miniature Computer, you will have to learn a new circuit to add to the AND circuit and the OR circuit with which you have already been working. This is called a NOT circuit. Instead of passing the current through when it is turned on, this circuit cuts off the current when it is put on.

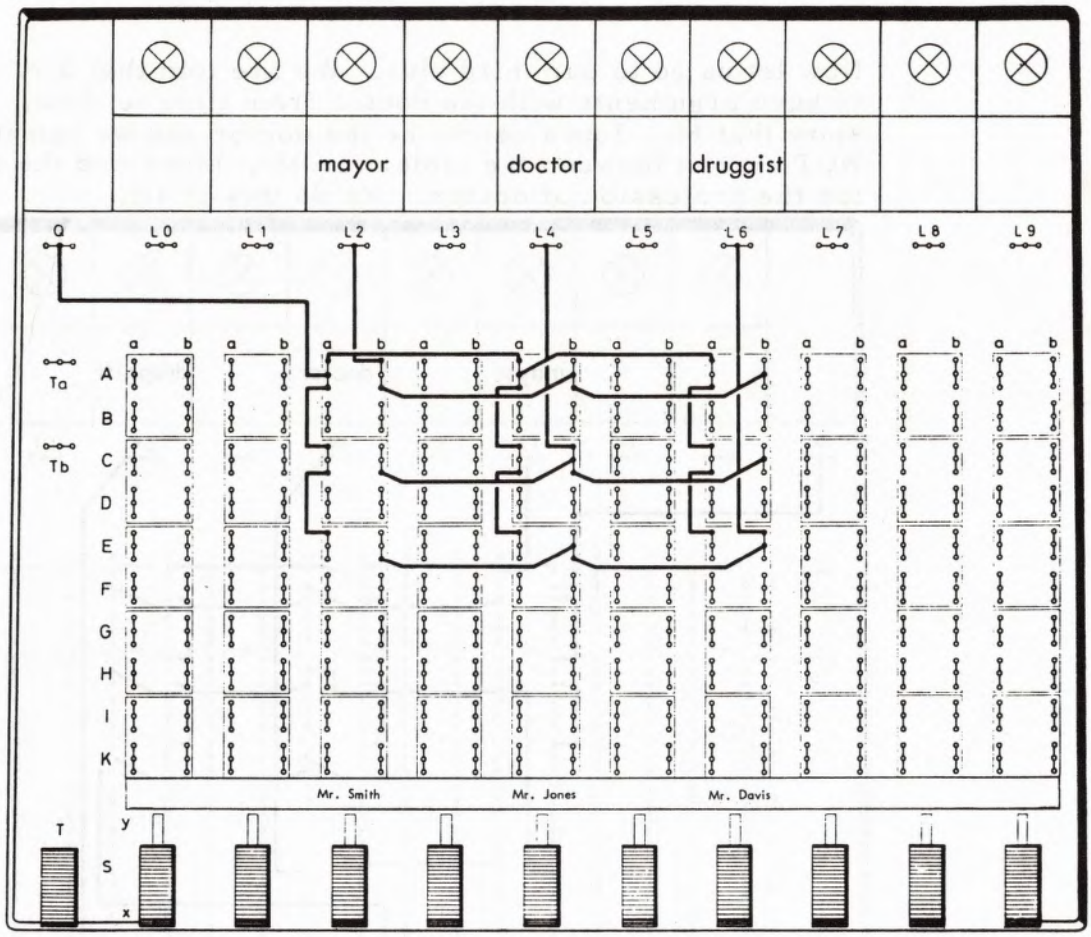
In the language of logic, a NOT circuit is represented like this:

In our work so far on our Miniature Computer, we have used the switches of rows A, C, E, G and I. To form NOT circuits, we use the bottom sets of switches B, D, F, H and K. For example, you can connect Bulb 2 to Q across D2 and when Slider 2 is in the OFF position, the bulb goes out. Thus, you can see that the NOT circuit works exactly in the opposite way to the circuits we have worked with up to now.



We will now see how we can use the NOT circuit in our problem with gentlemen Smith, Jones and Davis. First, follow the Wiring Scheme 21a. There is nothing remarkable in this scheme except from the point of view of logic. When you press the slider for any one of the three men (Mr. Smith, Mr. Jones or Mr. Davis), all three professions will light up at once. This is what happens when we try to solve this problem using only OR circuits.

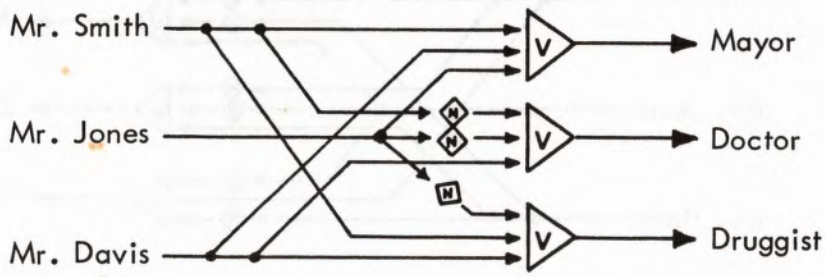
Now, let us try to introduce NOT circuits to this program; by



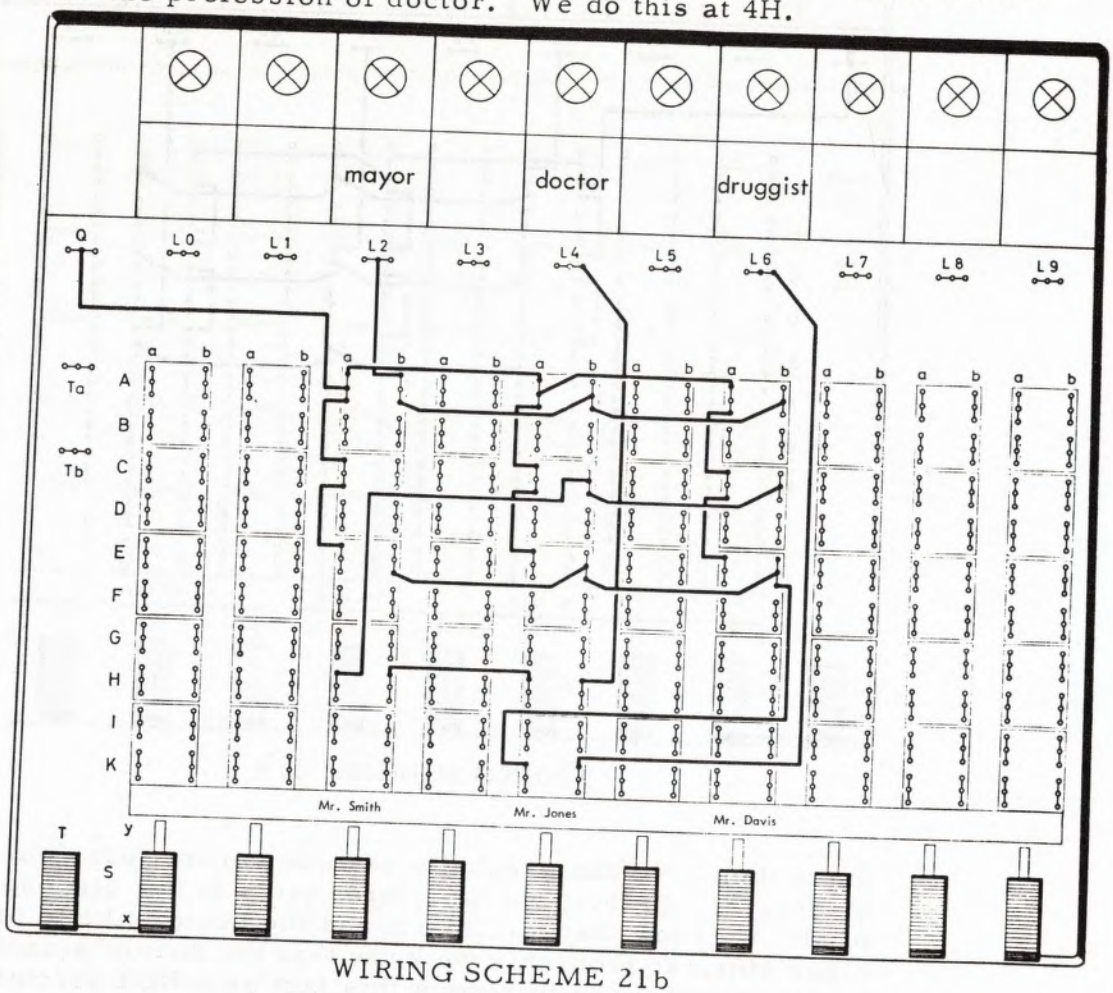
WIRING SCHEME 21a

using them, we can gradually exclude various possibilities one by one. To do this, we have three facts at our disposal. First of all, we know that Mr. Smith and the doctor like to tell each other stories; from this we know that the doctor's name cannot be Mr. Smith. We introduce this fact as a NOT circuit. Thus, we connect the wire that leads from the switch for Mr. Smith to the indicator for the profession of doctor through a NOT circuit, at H. Now when we push the slider for Mr. Smith, the indicator for the profession of doctor will not light up. This is already a start, since we have ruled out one profession for one man.

Let us consider the second statement. We are told that the druggist likes to play chess with Mr. Jones. Obviously the druggist cannot be called Mr. Jones. Consequently, we can separate Mr. Jones from the profession of druggist through a NOT circuit that we can apply at 4K. Now when we advance the slider for Mr. Jones, the indicator for druggist will not light up. We have ruled out another possibility.

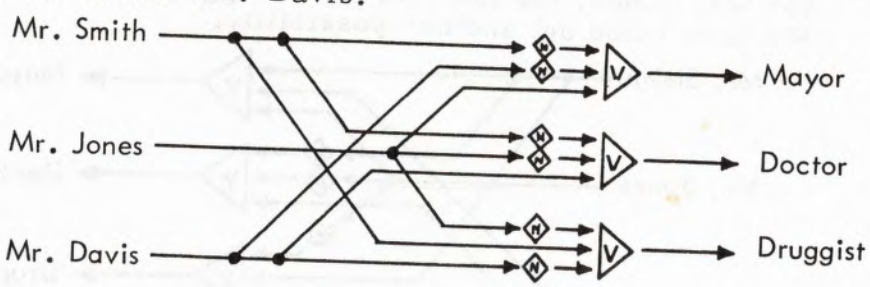


Now let us go to our third clue. We are told that Mr. Jones likes to have arguments with the doctor from time to time. Thus, we know that Mr. Jones cannot be the doctor and we can insert a NOT circuit between the slider for Mr. Jones and the indicator for the profession of doctor. We do this at 4H.

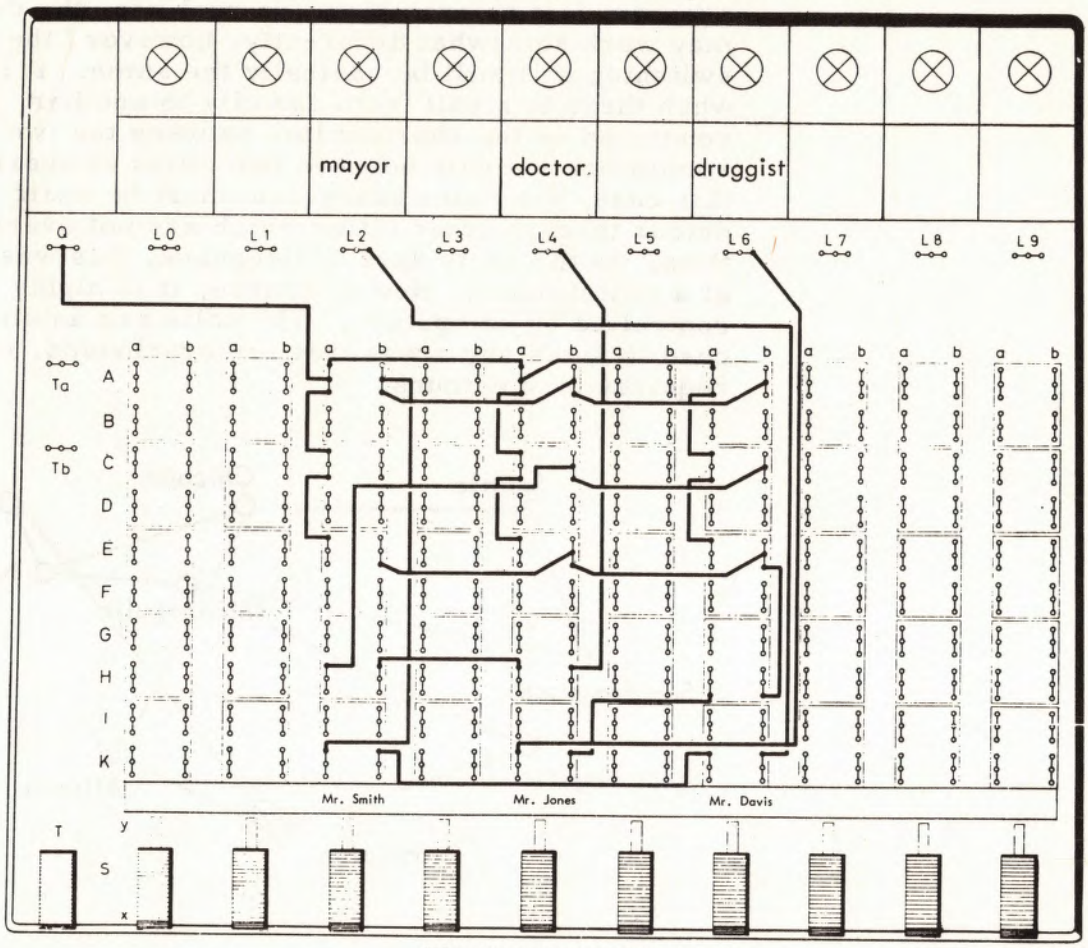


WIRING SCHEME 21b

Our program up to now shows us two interesting things: first that one can connect NOT circuits in a row just like AND circuits. The proof of this is in the connection between the NOT circuits at 2H and at 4H. The second thing that this program shows us, will be discovered by putting Slider 4 into its forward position. We learn the answer to the first part of our problem. Mr. Jones is the mayor. Now that we have found out that Mr. Jones is the mayor, we do not have to take this profession into consideration for Mr. Davis and Mr. Smith. Therefore, we can use 2 NOT circuits to separate the profession of mayor from the sliders of Mr. Jones and Mr. Davis.

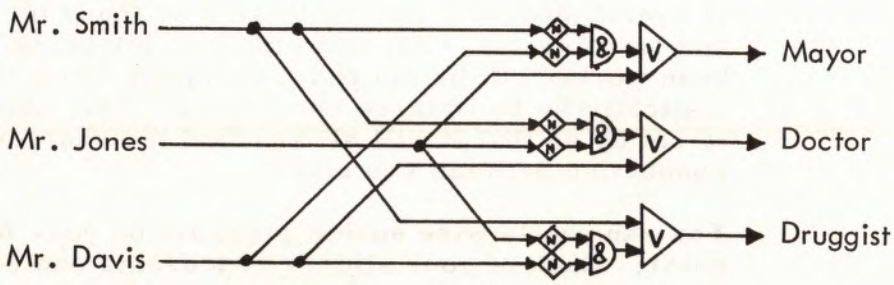


We can apply these at K2 and K6 in an AND circuit. Now with this addition, let us see if we can get a new result. By moving Slider 2 forward, we discover that Mr. Smith is the druggist. As a result, it is clear that Mr. Davis is the doctor. Let us finish our program by passing the connection between Mr. Davis and the profession of druggist, through a NOT circuit. If you look carefully at the diagram, you will find it. Now our program is perfectly set up and it looks like Wiring Scheme 21c.



WIRING SCHEME 21c

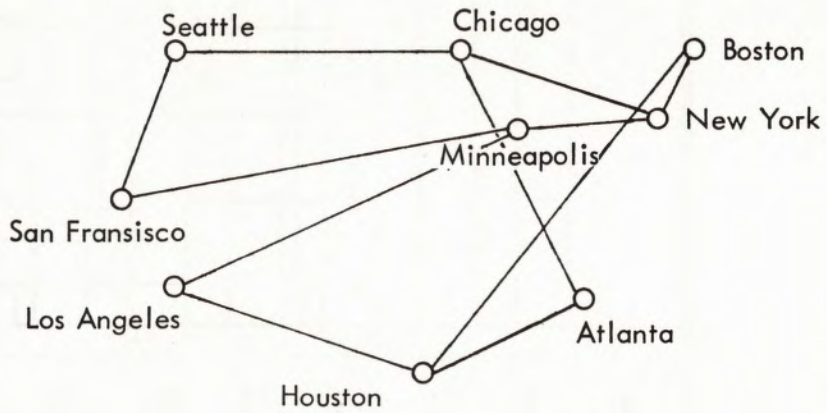
If we want to be more sophisticated, we can represent it like this:



PROGRAM 22 - TELEPHONE SWITCHBOARD

-Use transparent sheet No. 22.

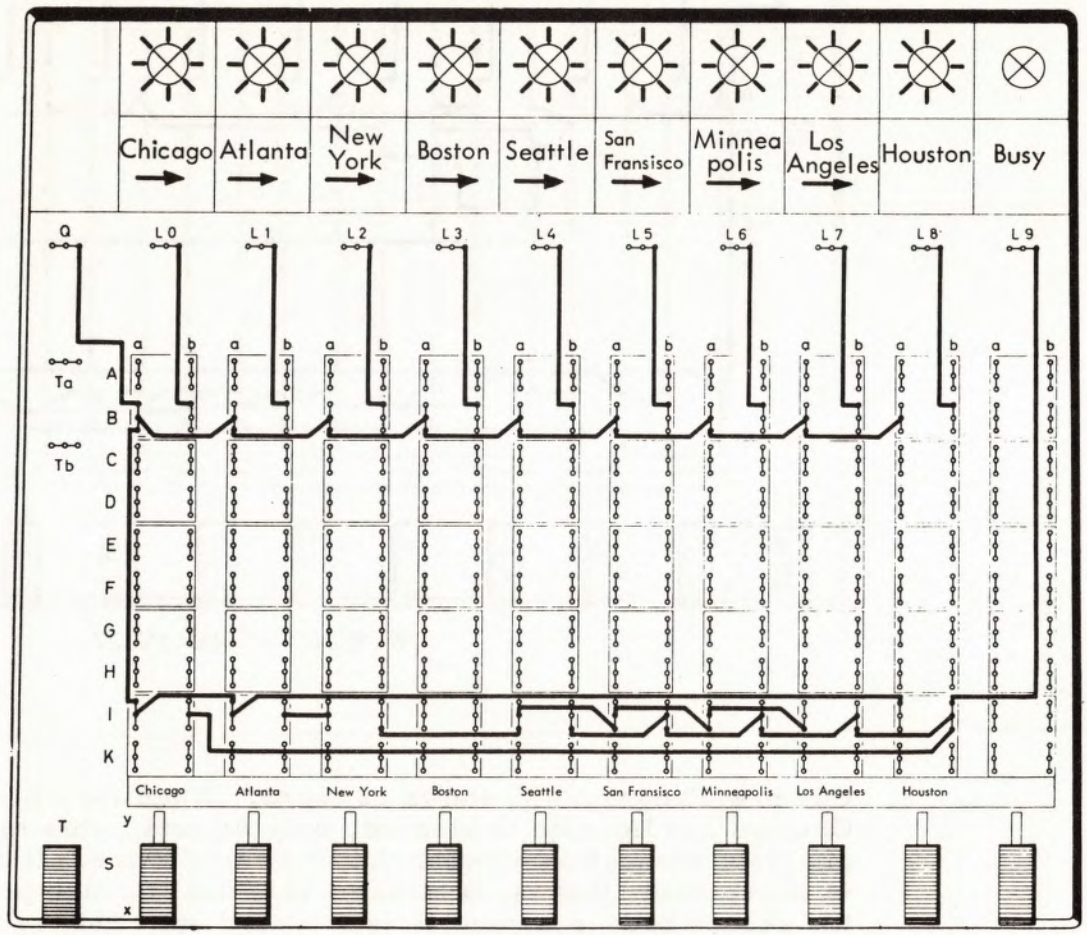
In this program we will try to imitate, in a very simplified fashion, how telephone company computers route long distance calls between cities. We have used the cities in our program just as examples. In real life, the connections may work somewhat differently, however, the principle of switching will still be basically the same. It is obvious that when there is a call from one city to another, it should be connected on the shortest line between the two cities; but sometimes the path between two cities is overloaded. In this case, the phone connection must be switched to take a detour through other cities which are not overloaded at this time. In the early days of telephone, this was done by hand at a switchboard. Now of course, it is highly automated and controlled by computers. The calls are automatically switched to avoid any areas that are overloaded, even if this requires long detours.



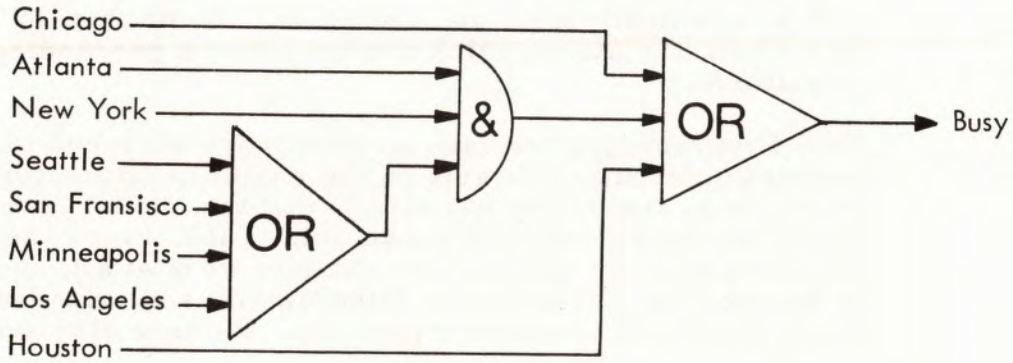
Let us look at our map. In this problem, we will attempt to route a call from Chicago to Houston. The shortest way of doing this would seem to be to route the call through Atlanta. But what if Atlanta is overloaded? In that case, we could reroute the call through Boston and New York. If Boston is overloaded, the call could still be made between Chicago and Los Angeles. All this efficient detouring has already been worked out by the phone company and is followed automatically by their large computers. This automatic system is set up so that it will always find the shortest and simplest connection between two cities.

You can easily wire such a program on your Miniature Computer. Each of your sliders represents one of the long distance cities. While testing the different possible detours, you show that a city is overloaded by setting the appropriate slider into its forward (y) position. Our circuit then immediately points out the next connection. We will work out such a

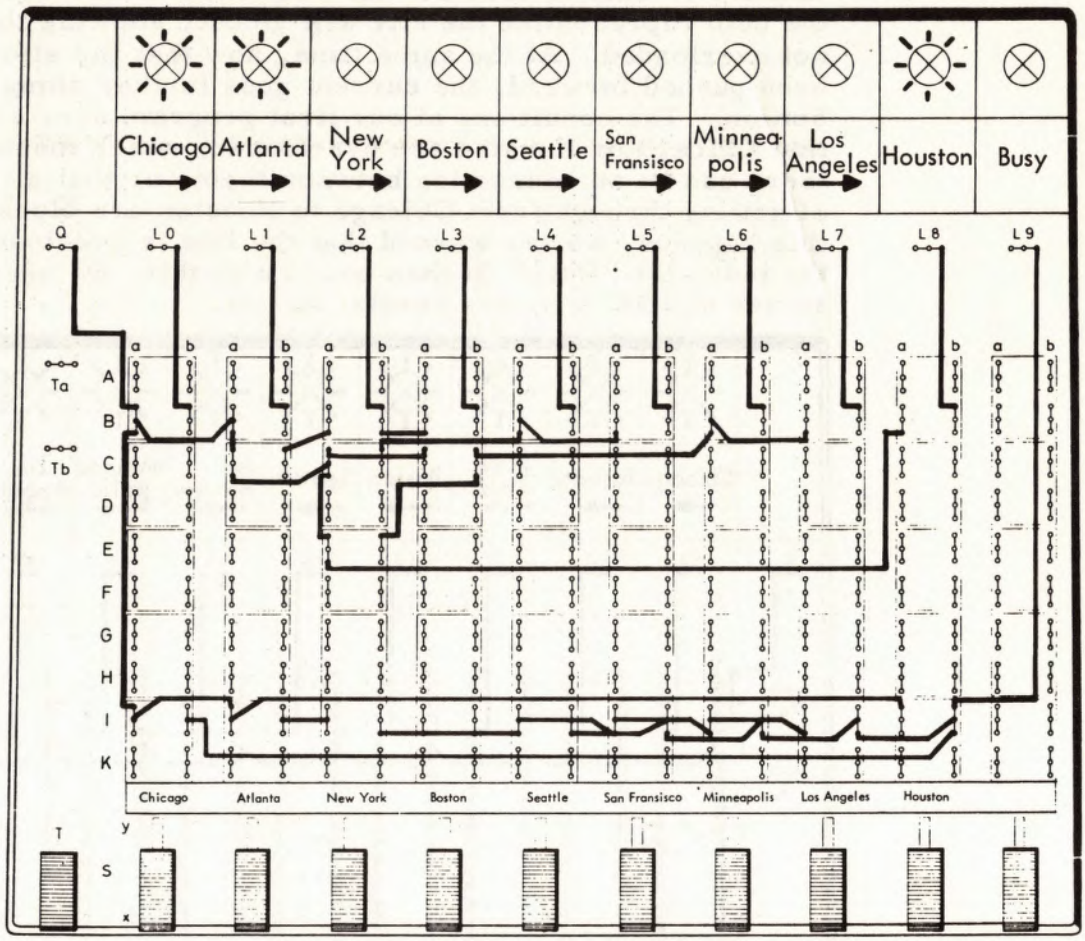
a program step by step, starting with Program 22a. This is the first stage. Every slider is connected by a NOT circuit to a bulb. This is done so that when the switch is set forward, the bulb representing the city will go out, showing that it is not overloaded. At the same time, now that the slider has been pushed forward, the current goes further along at the bottom. The conditions of our first program are: either that Chicago or Houston are overloaded, which means that there can be no connection between them; or that all the ways of getting through from Chicago to Houston are blocked. When this happens, we are warned that the line is overloaded when the indicator "Busy" flashes on. To do this, we need a series of "OR" circuits similar to this.



WIRING SCHEME 22a



This simple program shows us that the path is blocked, but it does not reroute the call along the two alternative routes previously mentioned.

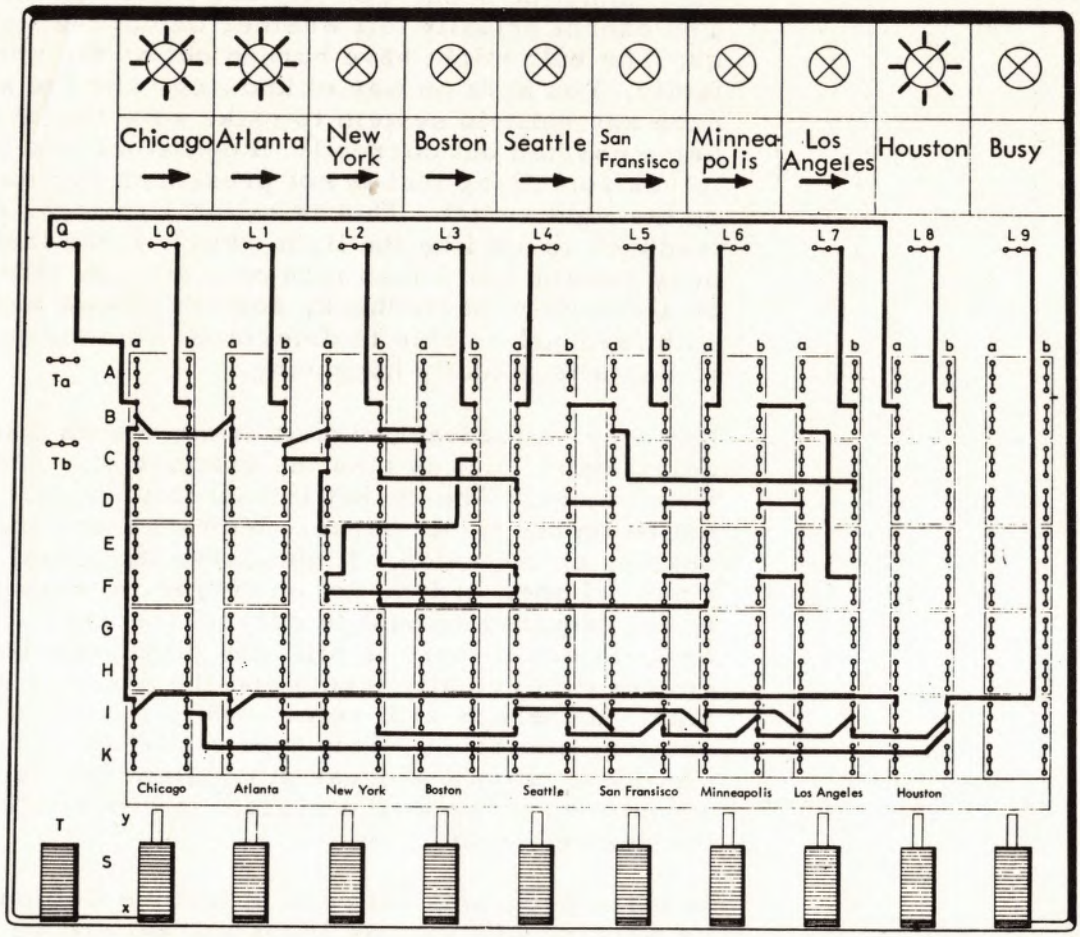


WIRING SCHEME 22b

Our new Program 22b shows us more. When the line between Chicago and Houston is blocked, only Atlanta lights up as the simplest connection between the other two cities. If Atlanta is overloaded (that is, if Slider 1 is in the forward position), the second path is immediately switched on. This is the route via New York and Boston. If Boston also happens to be busy, then the detour via New York, Los Angeles and Minneapolis will automatically light up, and so on. We do this by lighting up a chain of stations which are set off by a preceeding chain of stations.

With Program 22c, we have an even more advanced telephone exchange computer. It acts on the chains of connections in such a way, that if any one member of the chain is overloaded, the lights for the whole chain go out. Thus, every member of the chain must be open for the chain to work at all; if any one of Seattle, San Francisco or Minneapolis is overloaded, the whole chain will immediately go out. We have also set up the

wiring so that the bulbs can only light up when the station for Atlanta is overloaded. Actually, real computers which control the routing of long distance calls will first test the stations along the simplest route, and then test the stations of succeedingly more difficult routes until they have found the simplest, free route.



WIRING SCHEME 22c

Our Miniature Computer can now do the same type of things. First, all the sliders must be turned to the backward position. In this position, the connection between Chicago and Houston is connected by that of Atlanta. If Atlanta is busy, that is we switch Slider 1 to its forward position, the next path automatically lights up. If a station there is overloaded, which we show by putting one of the sliders in its forward position, our Miniature Computer will immediately show the next path. Or if it turns out that there is no possible connection at this time, it will in the end show that the line is completely overloaded.

CYBERNETICS

Cybernetics is the study of regulation as contrasted with direction. Let us give two examples. An example of direction is to turn on a switch connected to an outside light bulb that cannot be seen. You flip the switch to the ON position. You cannot actually tell whether or not the light is actually on. The bulb might have burned out or the wiring might be faulty. You have no way of knowing, and the switch does not have any built-in system to make sure that its purpose is being carried out correctly. Regulation is different. It involves something that is not present in our example of turning on the light switch. This is called feedback. A switch with feedback is not like the light switch of the last example which does nothing more than turn on a circuit. After you switch on a circuit with feedback, something new happens. A circuit with feedback is able to determine whether or not the process it starts is actually happening.

However, a feedback circuit is more than this. It involves adjustment. Let us take the example of a rocket being shot to the moon. The rocket is shot into space. It may or may not be on the right course. With its instruments it takes a bearing on the moon. It compares its actual course to the required one. If it is not on course, it makes an adjustment of the required amount to correct the error. Then, it checks again to see if there is still any difference between the course it is actually taking and the one it is supposed to take. If there is still some error, it makes another correction, and so on. This type of circuit is known as a control loop. It is a process of constant feedback and constant adjustment throughout a system in order to keep it within the desired limits.

In these programs with our Miniature Computer, we will be working with control loops and we will be learning the many areas in which they are used in real life.

Everyone is constantly coming across these control loops without being aware of them. The most common control loops are for temperature. Think of a central heating unit for a house. This self-regulating cycle is made up of two essential units: a furnace to provide the heat, and a thermostat to provide the control. The thermostat is set at the desired temperature and it is constructed to allow a certain number of degrees of temperature above or below the temperature set on its dial. As soon as the temperature of the room rises above this level, it connects a circuit in the thermostat that turns off the furnace. This is an example of feedback. Now that the furnace is turned off, the room begins to cool. As this happens, the thermometer in the thermostat falls until it reaches the lower limit set in the thermostat. At this point, another switch in the thermostat is closed and it now automatically turns on the furnace. Now

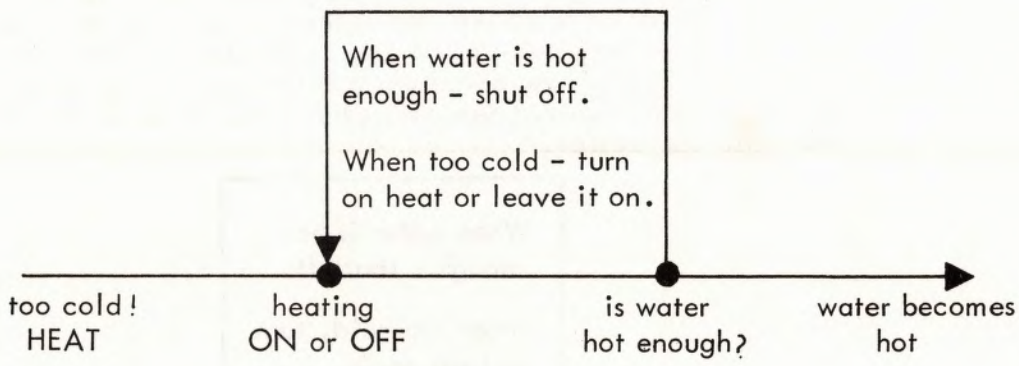
the room will heat up until the high point is reached again. With this self-regulating cycle which controls the furnace by using feedback in the form of temperature readings, the room can be kept at a stable temperature that does not vary too much above or below the one you set on the thermostat dial. This is not only a control loop, but also a self-regulating cycle since the apparatus of the whole furnace and the thermostat alone can control the whole process.

In our programs, we will start with control loops that deal with temperature. Later on, we will learn about control loops in other areas. You may be amazed at all the areas of life that involve these loops.

PROGRAM 23 - A WARM BATH

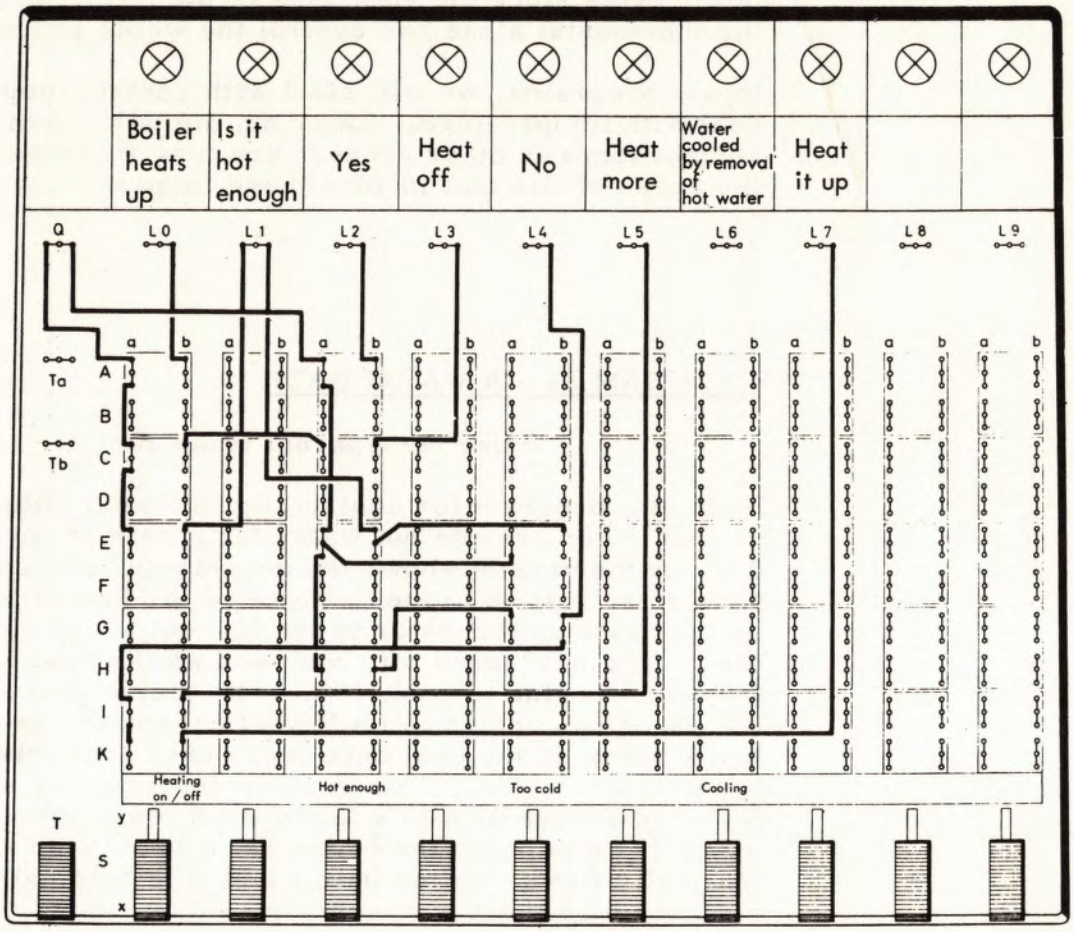
-Use transparent sheet No. 23.

Here is a program for controlling the temperature in a boiler, so that it can provide hot water for a bath at any time of the day. A thermostat shows the acceptable temperature of the water, just as in the example of the heated house. The boiler heats up the water to the highest acceptable temperature. The heat turns off. As soon as the heat turns off, the water begins to cool. When the thermostat senses that the water has cooled to its lowest acceptable level, it immediately turns on the heat once more until it reaches the right temperature again. This goes on continuously so that the water is always kept at a fairly even temperature. When water is taken from the boiler for a bath, cold water automatically flows into the boiler and it is automatically heated until the high point of the thermostat is reached again. From then on, it too is kept at a fairly constant temperature. We have worked out a diagram showing how the boiler is kept at an even temperature. This is a schematic representation of this control loop.



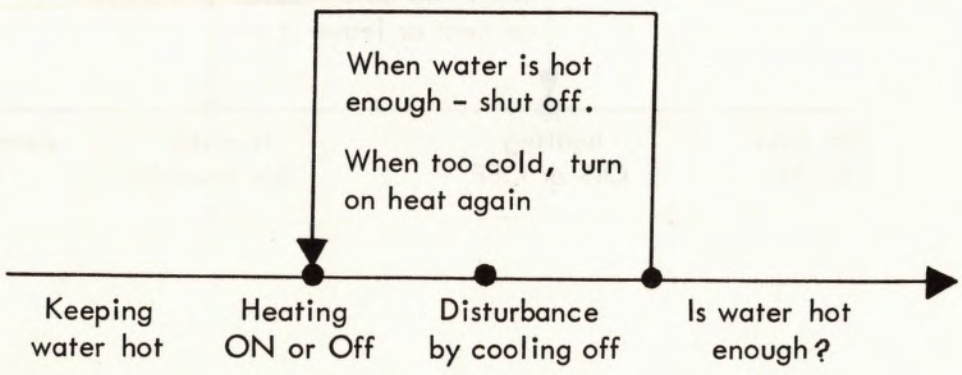
With Wiring Scheme 23a, you begin to regulate the heat by moving Slider 0 forward. Depending on the water temperature, turn your switch on Slider 2 or 4 ("Hot enough" or "Too cold").

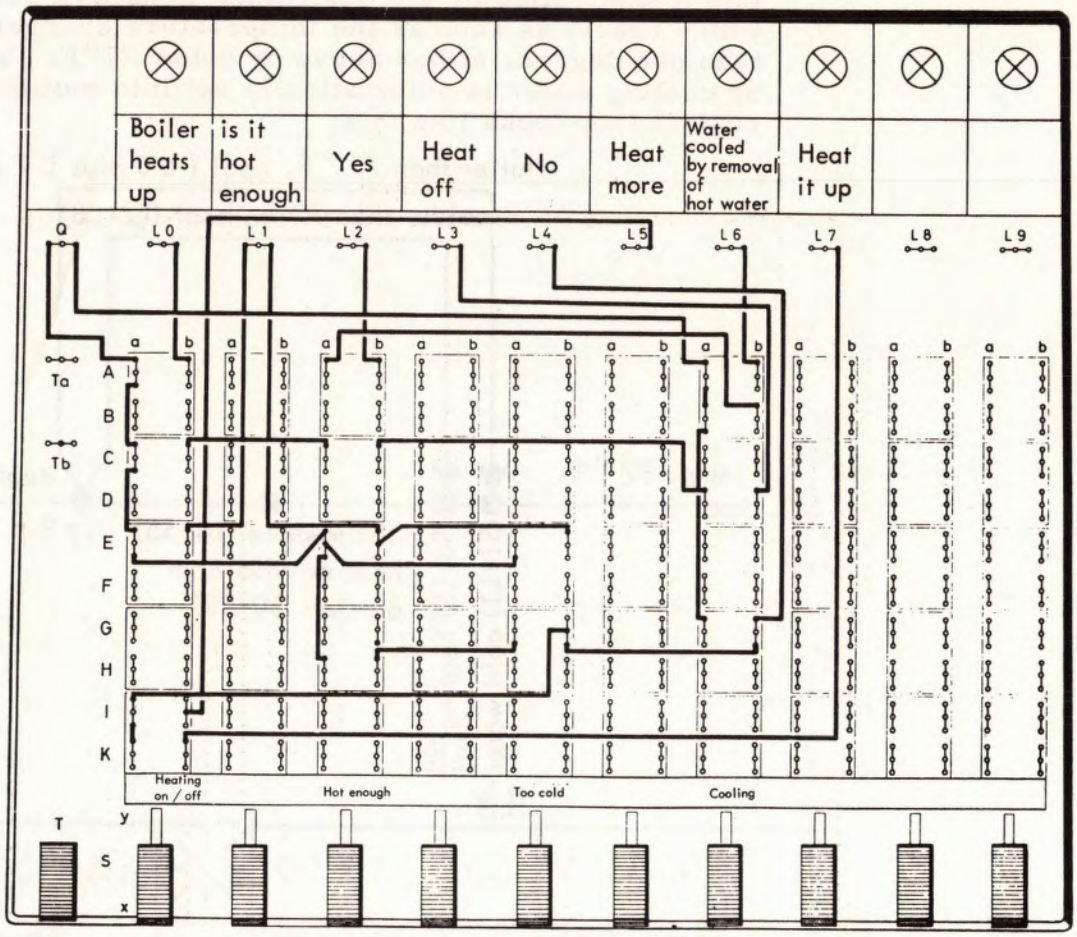
Except for this, you regulate the heat according to the indications given on the Output Section of the Computer. Note



WIRING SCHEME 23a

that Slider 0 is on or off, depending on whether it is in the x or y position. Up to now we have looked at only the heating procedure of the boiler. Now let us add another complication by assuming that the water in the boiler cools off for some reason, whether because the bathroom is very cold or because the hot water has been used and replaced by cold water. The regulation cycle now becomes a bit more complicated because there is an additional factor added - the cooling off of the water in the boiler.


When water is hot enough - shut off.
When too cold, turn on heat again
Keeping water hot
Heating ON or Off
Disturbance by cooling off
Is water hot enough?



WIRING SCHEME 23b

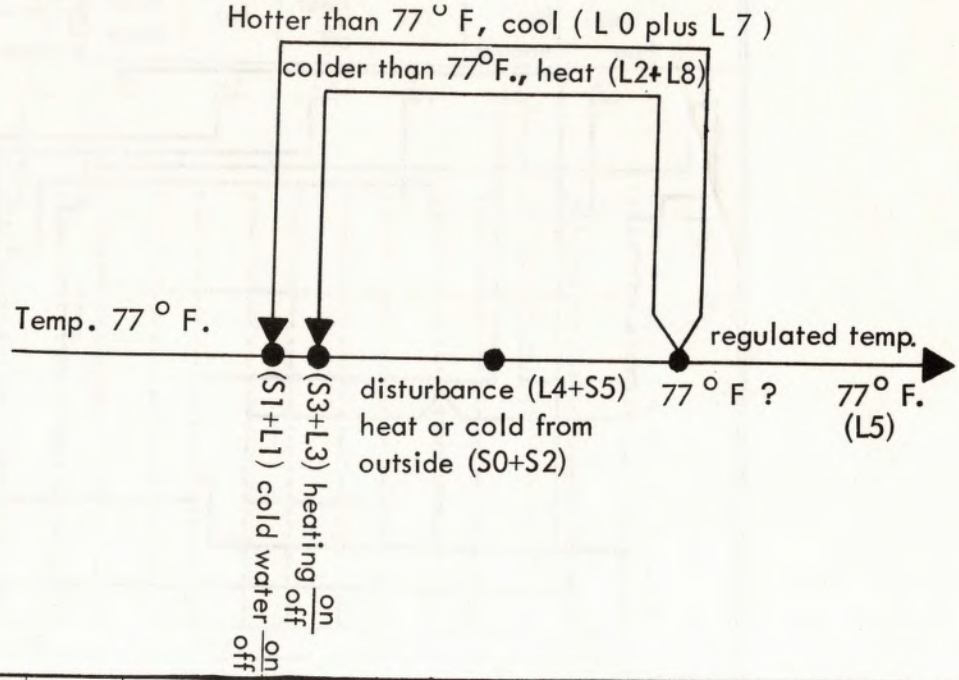
This small, but very important addition can be easily shown on your Miniature Computer, as in Wiring Scheme 23b. Although in Wiring Scheme 23a, we have looked only at an example of cooling down by removing hot water, the control procedure works exactly the same way in every type of cooling off. Whenever the cooling has been compensated for by renewed heating, you must not forget to set Slider 6 back to x. Now if you connect a thermometer to your Miniature Computer and have it controlled by a heat switch, you could always keep your bath water at the right temperature with the help of the control program.

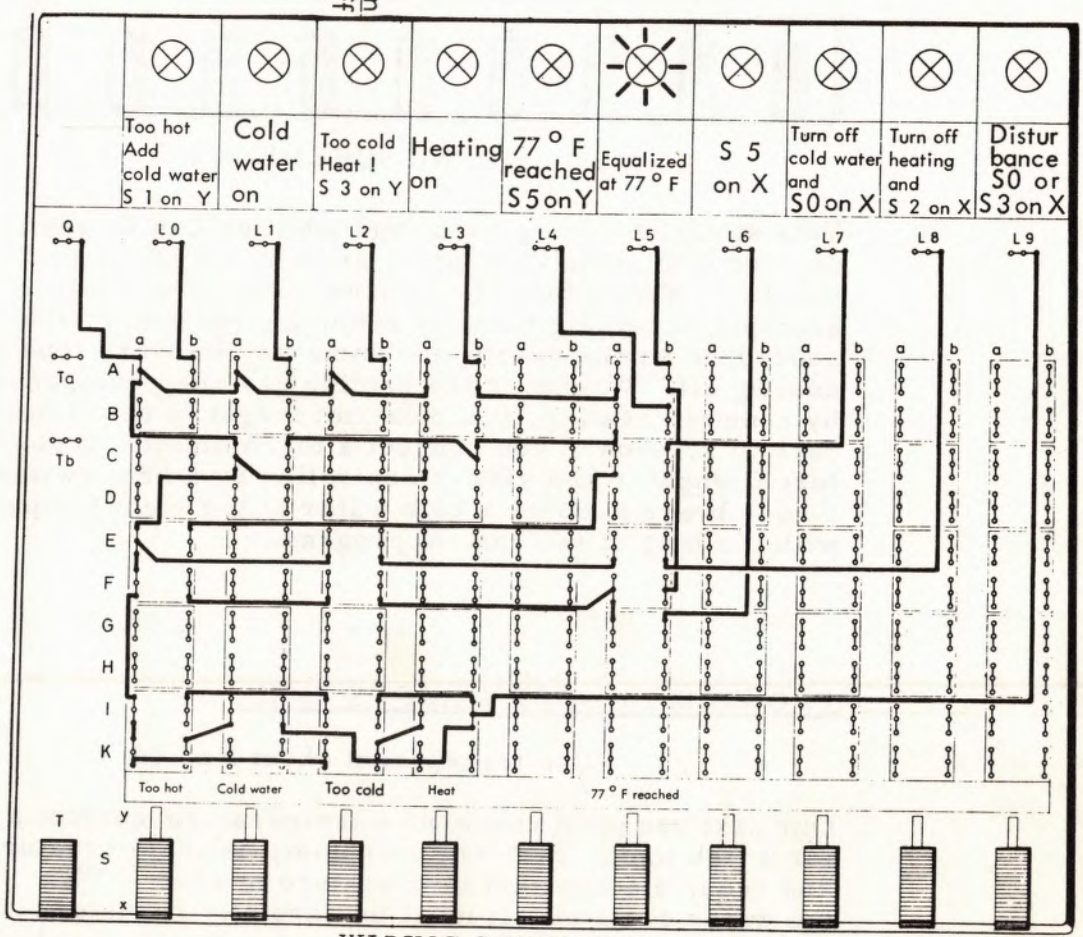
PROGRAM 24 - FOR A HAPPY FISH

-Use transparent sheet No. 24.

Our next program concerns a temperature-control system for a fish tank. Fish are extremely delicate creatures and require a constant temperature of about 77°F. If the water cools off, it must be carefully reheated. If it becomes too warm - perhaps because the sun is shining on the tank - cooling water must be added. In practice,

this is controlled by a temperature-sensitive device which reacts as soon as the temperature changes more than one degree, either above or below 77°F. The heating or cooling water is automatically set into motion, and the control loop looks like this:


Hotter than 77 ° F, cool (L 0 plus L 7)
colder than 77°F., heat (L2+L8)
Temp. 77 ° F.
regulated temp.
(S1+L1) cold water on/off
(S3+L3) heating on/off
disturbance (L4+S5)
heat or cold from outside (S0+S2)
77 ° F ?
77° F. (L5)



WIRING SCHEME 24

This interesting control loop can be programmed on the Miniature Computer. We have noted the points where we have to think of the various sliders and bulbs. The temperature thermostat (the device which is sensitive to temperature) is in this case situated on Sliders 0 and 2; it must react to water that is either too hot or too cold. The cooling or heating water has to be switched on accordingly until a temperature of 77°F. has been reached once more. In Wiring Scheme 24, you will notice in the left hand corner a small program area which does not in itself have anything to do with the temperature control. It shows when the indicators for "Too hot" or "Too cold" are lighting up at the same time (which cannot actually be the case), or when the cooling or heating water is turned on without the thermostat. This small technical safety device is part of many pilot systems and is usually connected with a warning light or bell, so that the supervisor can take preventive measures before anything disastrous happens.

PROGRAM 25 - BODY HEAT

- Make your own transparent sheet for this program.

The following programs are mainly concerned with circular structures, such as the inner construction of machines, animals, economic systems, or push button factories. For this purpose, our Miniature Computer likes to build models. For example, it invents artificial animals which are really small machines but which sometimes act like living creatures by imitating the behavior of animals.

Wiring Scheme 24 shows us how the water temperature in a fish tank can be kept at exactly 77°F. This is exactly the same problem that exists in the human body. The human body must constantly work to keep its temperature at 98.6°F. This is done in a way quite similar to the one we used in our fish tank. If it becomes too cold, the body stays at the proper temperature by burning up more nourishment. When there is too much heat, the body produces perspiration which evaporates and thus lowers the body temperature. All we have to do now, is to put on a new transparent sheet and use Wiring Scheme 24 which does not change at all.

PROGRAM 26 - THE HUMAN EYE

- Make your own transparent sheet for this program.

We could also use the same wiring shceme (Wiring Scheme 24) as a control system for a camera. Let us assume

that the exposure time (perhaps 1/50 of a second) is unchangeable. Now you adjust the opening according to the amount of incoming light. The light intensity is set on Sliders 0 to 3 and the aperture is regulated on Sliders 6 to 9. The wording for the transparent strip is "Choose larger aperture" or "Choose smaller aperture".

A camera with an aperture which is adjusted according to the strength of the light, is similar to the human eye. When the incoming light becomes stronger, the iris in the human eye reacts by contracting and the pupil becomes smaller.

This can be imitated with Wiring Scheme 24. In this case, the light intensity is set on the left side. The diameter of the pupil is set on the right side. The light symbols once more tell us whether the pupil has to contract more, whether it can open further, or whether it has the correct diameter. The control mechanism constantly checks whether or not the diameter of the pupil is appropriate to the intensity of the light, and it adjusts the pupil accordingly. This sometimes takes half a second. When a person suddenly enters a well-lit room from a dark room, the control mechanism does not act instantly. The pupil takes a moment to contract and this often results in a moment's blindness.

PROGRAM 27 - HEATING A HOUSE

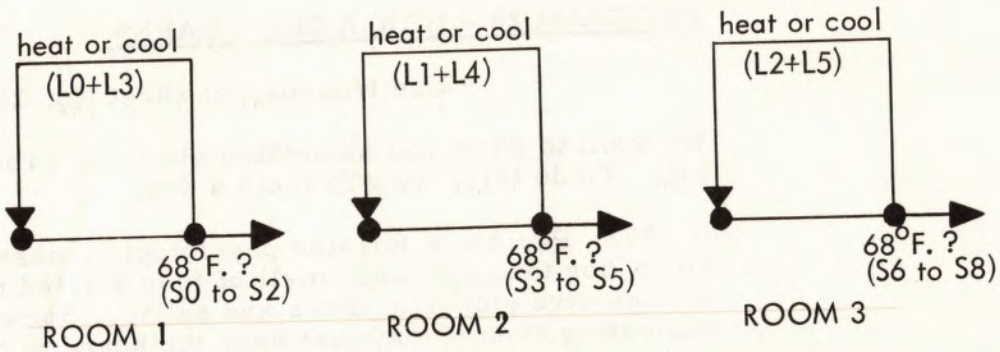
- Use transparent sheet No. 27.

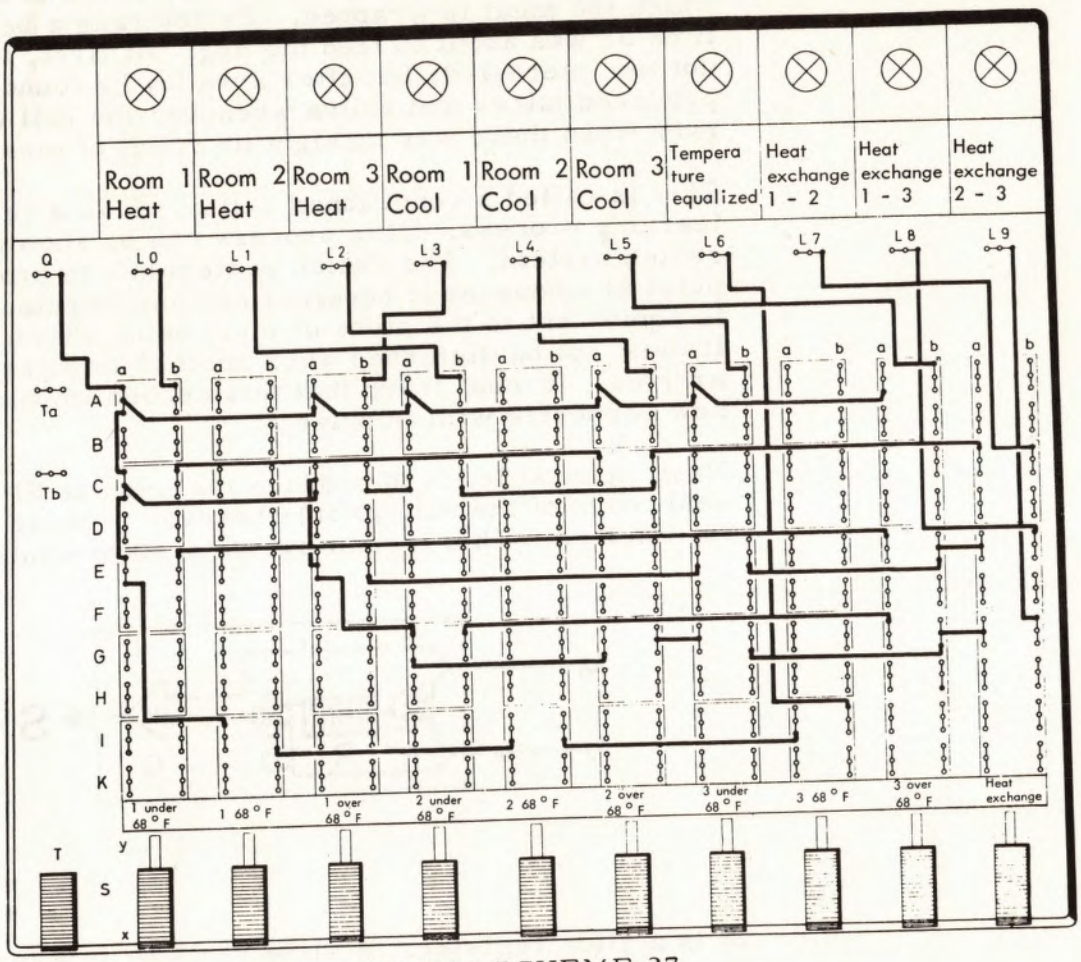
We can now apply our temperature-control system to a house with three rooms. In Wiring Scheme 27, we have rooms 1, 2 and 3. On Sliders 0 to 8, we feed in the present temperature of these rooms (below or above 68°F.). The ideal temperature of these rooms is 68°F. Our Miniature Computer tells us instantly which rooms we have to heat and which we have to cool. If the temperature reaches the required level, the bulbs indicating heating or cooling are switched off.

As soon as all the three rooms have been brought to the required temperature of 68°F., the "Temperature equalized" indicator lights up. Our problem, however, becomes really complicated when we adjust Slider 9.

In modern heating methods, there is an ideal way to save energy: the heat exchanger. With this technique, heat is diverted from overheated rooms to cold rooms. This is what happens when we bring Slider 9 into play. Then Lights 7 to 9 show us which rooms need heat exchange. This program has several control loops that are closely connected to each other.

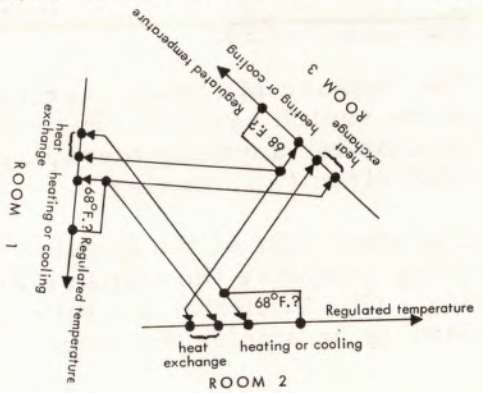
First, we have the three control loops for rooms 1, 2 and 3.





WIRING SCHEME 27

If we use the heat exchange process, there are three more control patterns, for now each control loop can influence the other two. The complete control system looks like this:



PROGRAM 28 - HOW A DOG LEARNS

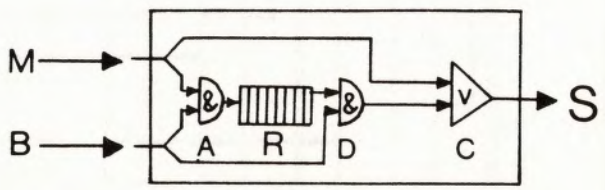
- Use transparent sheet No. 28.

We want to show you something about the process of learning. To do this, we will train a dog.

In 1898, Pavlov, a Russian physiologist, observed that for a dog the sight and smell of food started the production of digestive stomach juices and saliva. Some dogs start salivating even if they just hear the sound of the paper in which the meat is wrapped. Pavlov rang a bell every time he was about to feed the dog. At first, the dog was not too interested; but after a while, he found that the dog produced juices and saliva whenever the bell was sounded, even when there was no sight or smell of meat.

This is called a conditioned reflex. It is a very simple learning process. This process can be shown as a logical switch system. The switch system has to produce an impulse S1 whenever it receives another impulse M, which is equivalent to the piece of meat being shown to the dog. It must not be disturbed when another impulse B, a bell, arrives. It must learn that this second impulse is somehow connected with impulse M.

Next, it must learn to produce the impulse S1 (which is the equivalent of the salivation) whenever impulse B (the bell) is received. This logical switch system would look like this:



R is a slide register. A slide register is a type of storage or memory area in which the entire supply of electric impulses is moved one compartment ahead as another impulse is added at the rear. In this way, the impulses are pushed through the register by pressure from the back.

Experiment 28a

The impulse M (meat) arrives. It divides itself at the entrance of the switch system. Part of it goes to A but is unable to accomplish anything there, as the second impulse of this AND circuit is missing.

The other half of the impulse M goes to the OR circuit at C which produces the impulse S1, the salivating effect of the dog. Our artificial dog thus produces the digestive juices as soon as he smells the meat.

Experiment 28b

Add impulse B (the bell) to impulse M (meat). In other words, hold the meat in front of the dog's nose and ring the bell at the same time. Part of impulse M passes directly through C. The dog reacts and the digestive juices flow.

The second part of impulse M goes to point A, and AND circuit, where it joins a part of impulse B and both move past point A together. They now form a common impulse which forces its way into the slide register. Whenever there are two impulses reaching a point at the same time, like meat and bell, they produce one effect.

Nothing else happens at this time. Actually, the dog has already learned something; he just does not know it yet.

What has happened is now repeated several times. M (meat) and B (bell) always arrive at the same time and pass point A. Slowly the slide register is filled and slowly the conditioned reflex is stabilized. Finally, the slide register becomes completely full. Now the learning process is finished.

You can continue to start both impulses M and B simultaneously. Nothing new happens. The slide register has found impulses pushed in at the back and lets go of others at the front. They meet at point D with part of the impulses of B; and pass together through D to the OR circuit at C; and do nothing there that could not be done by M alone, which is to get the dog to salivate (S1).

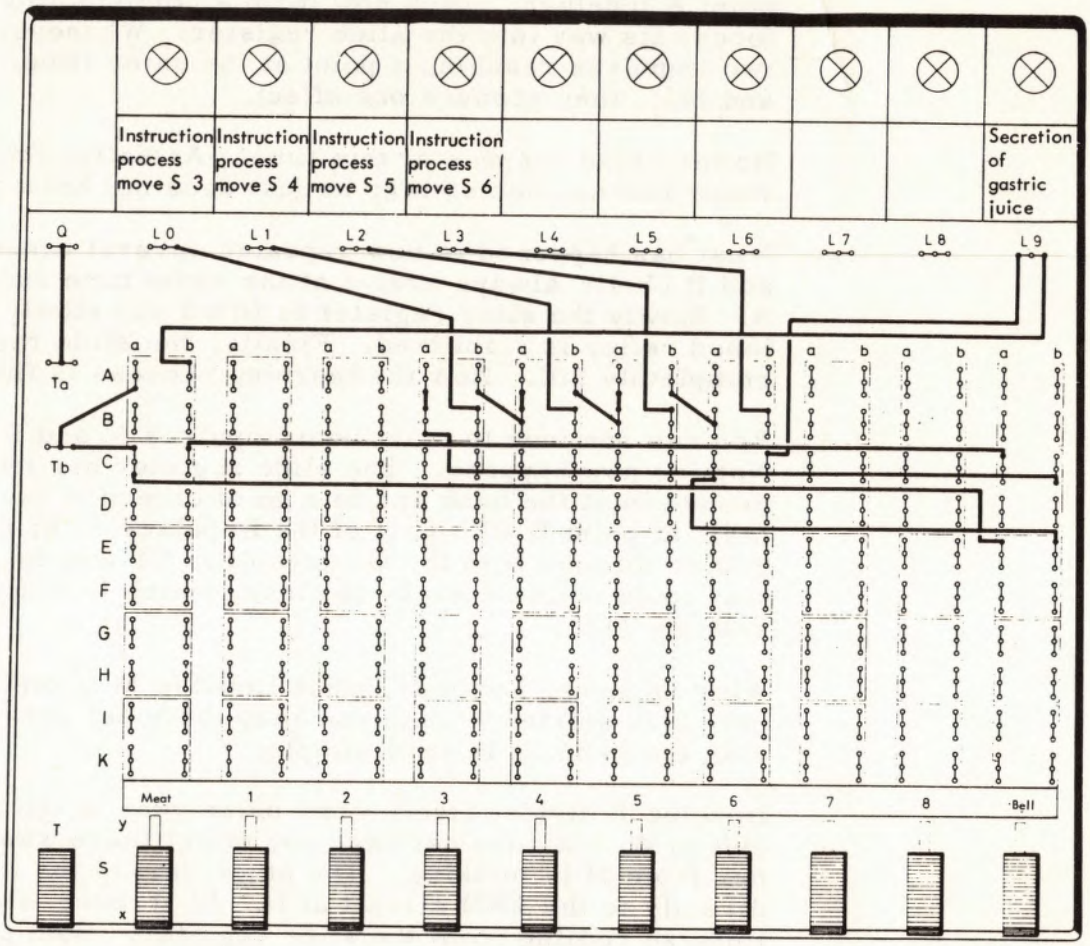
What happens, however, when impulse M is omitted and only B is started - when you keep the meat away and only ring the bell? It is very simple.

Impulse B divides itself - one pulse runs to the AND circuit at A, but does not have any effect there when its partner from M is missing. The other part of the impulse goes directly to the AND circuit at D. At D there is already an impulse coming from the slide register. Both pass together and go directly to the OR circuit at C and affect the impulse S 1. The bell alone has been enough to get the stomach juices started.

We can reproduce all this on our Miniature Computer, on Wiring Scheme 22. Here we have programmed Pavlov's dog exactly according to the switch plan we have outlined above. Unfortunately, unlike Pavlov, we do not have a slide register. You will have to take over this function yourself using Sliders 3 to 6.

Let us see how our good dog reacts as we slide Slider 0 (meat) from x to y and push button (T) is pressed. Light 9 lights up. This is a clear case: the dog gets his food and the digestive juices flow. If you now also let a bell ring and set Slider 9 at the same time as Slider 0, then when you press push button (T), not only does the reaction for the stomach juices light up but so does Light 0.

First set Slider 3, since it is the first impulse which sets itself into the dog's mind. Our electronic dog needs 4 of these learning processes before he will have learned it. After the fourth time, all you have to do is ring the bell and set Slider 9, Slider 0 may remain on x since the stomach juices now run without it. Our dog has learned something. To say it is a different way, our Miniature Computer has learned something. Impressive, isn't it?


Instruction process move S 3
Instruction process move S 4
Instruction process move S 5
Instruction process move S 6
Secretion of gastric juice
Q
L0 L1 L2 L3 L4 L5 L6 L7 L8 L9
Ta
Tb
A B C D E F G H I K
Meat 1 2 3 4 5 6 7 8 Bell
T
y S x

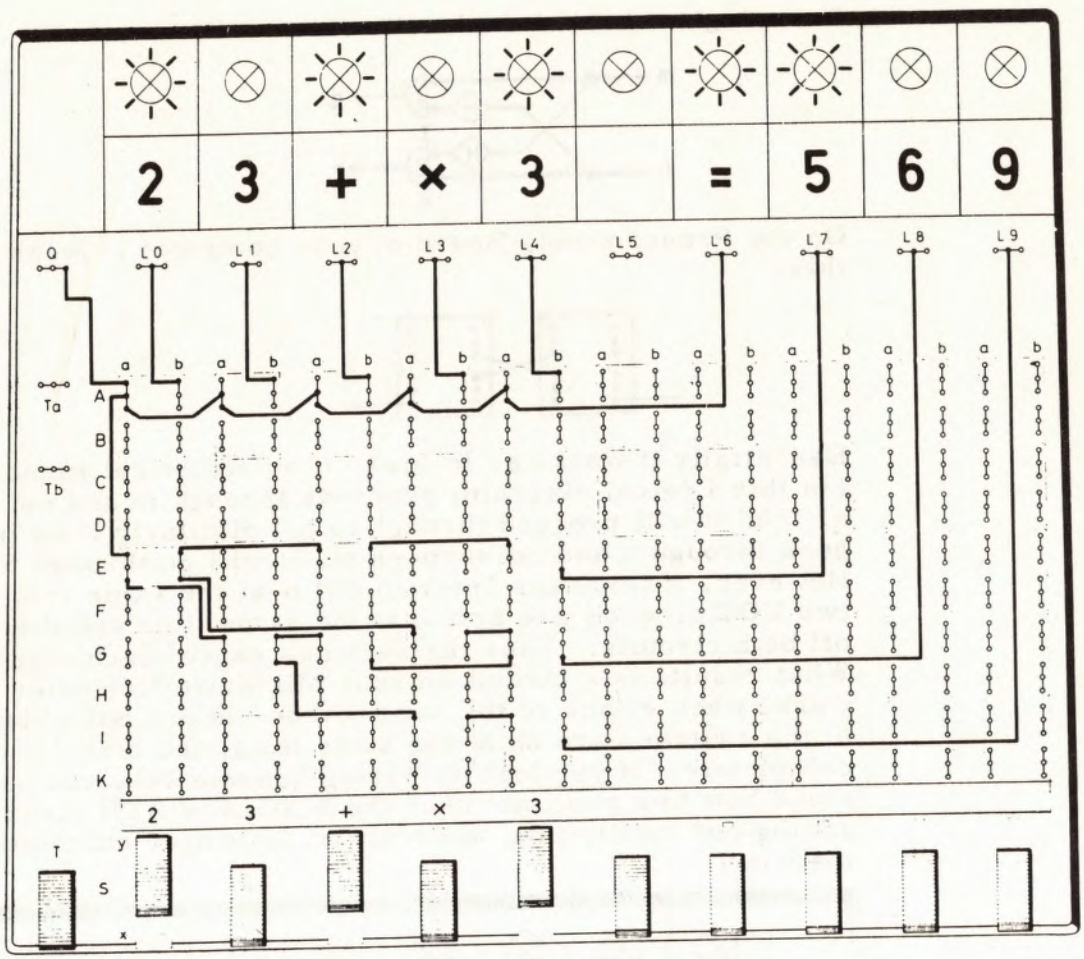
WIRING SCHEME 28

MATHEMATICAL CALCULATIONS

PROGRAM 29 - THE CALCULATOR

-Use transparent sheet No. 29.

It is not too difficult to convert your Miniature Computer into a simple desk calculator. You have already been working the basic calculating device we will use: the AND circuit. With the AND circuit we not only can add, but with a small adjustment, we can also multiply.



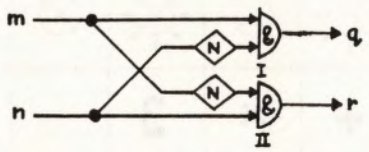
WIRING SCHEME 29a

Notice that in this program, the wiring for multiplication is exactly the same as that for addition - an AND circuit. The computer has "memorized" some addition and multiplication tables and finds the sum or product of two figures just as you would in looking them up in a table.

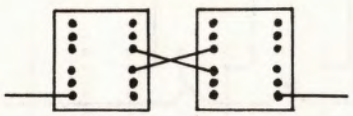
Also notice that several OR circuits must be used. Whenever we have several operations leading to the same answer (i.e. 2 x 3 = 3 + 3 = 6), we must connect the different paths with OR circuits. Our little calculating program has a few inconveniences. If you push Sliders 1 and 2 or 3 and 4 at the same time, you see several results appearing. This would be confusing to anyone who did not know the answer before he started. In the design of large computers, these errors are very important. In electronic bookkeeping, for example, they could upset weeks of calculations. For this reason, we must use NOT circuits to make sure that the computer does not follow these false paths.

We will now use a new circuit which will allow us to greatly improve the accuracy of our calculating programs by ruling out many false paths in our wiring. We call this the SINGLE-PATH circuit and it is basically made up of two AND circuits connected by two NOT circuits.

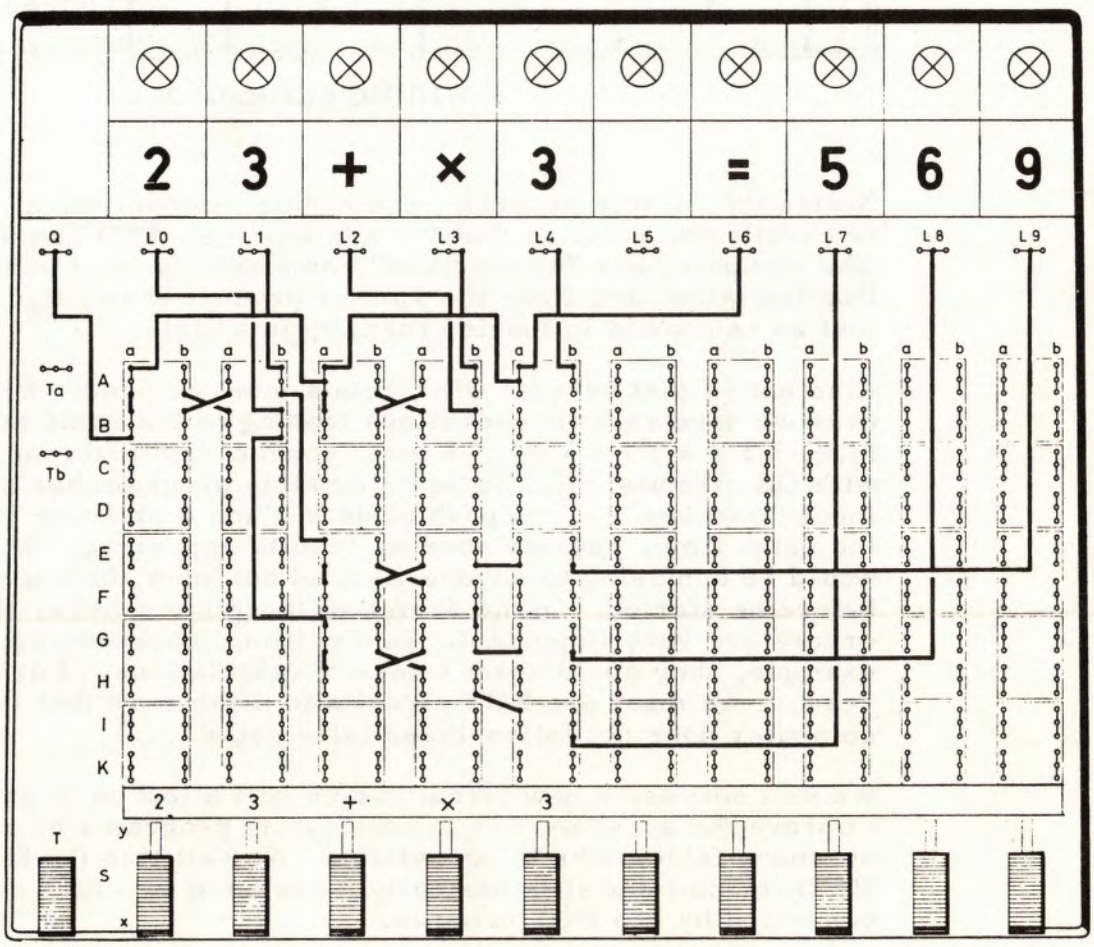
In its logical form it looks like this:



On the Programming Board of your computer it looks like this:

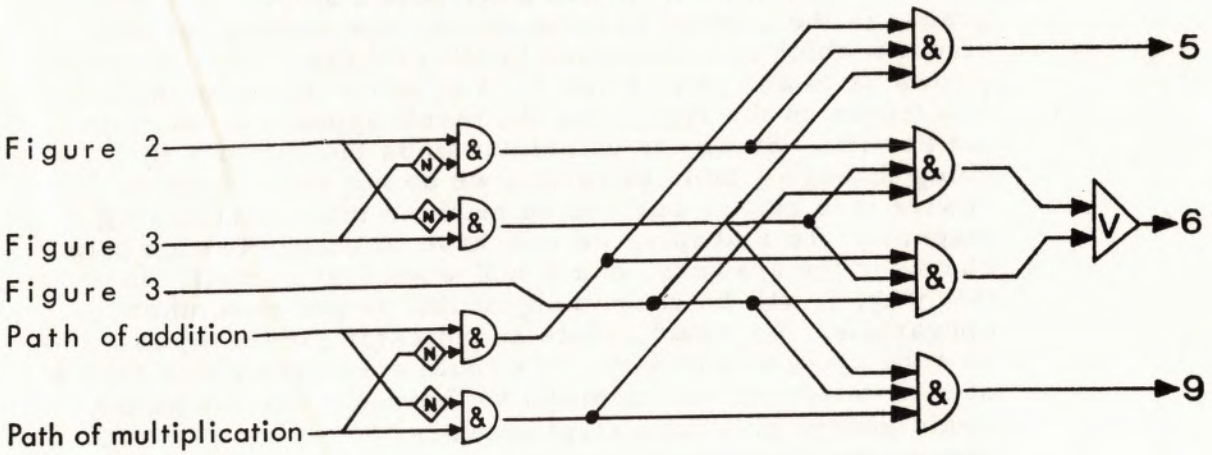


Essentially it works as follows: if an electrical impulse (in this Logical diagram) proceeds through m and not through n, then it will proceed through to q. Similarly if an impulse goes through n and not through m, it will go through to r. However, if impulses enter m and n at the same time, the two NOT circuits are active at the same time and they cut off both circuits. Thus the switches cancel each other out. What results is a circuit on your Miniature Computer that works when either of the two switches is on; but when two of the switches are on at the same time, the circuits cut out completely. If you look at Wiring Scheme 29b, you can understand how this program uses the SINGLE-PATH circuits for adding and multiplying to create an improved calculating machine.



WIRING SCHEME 29b

This is how we would show it in a logical scheme:

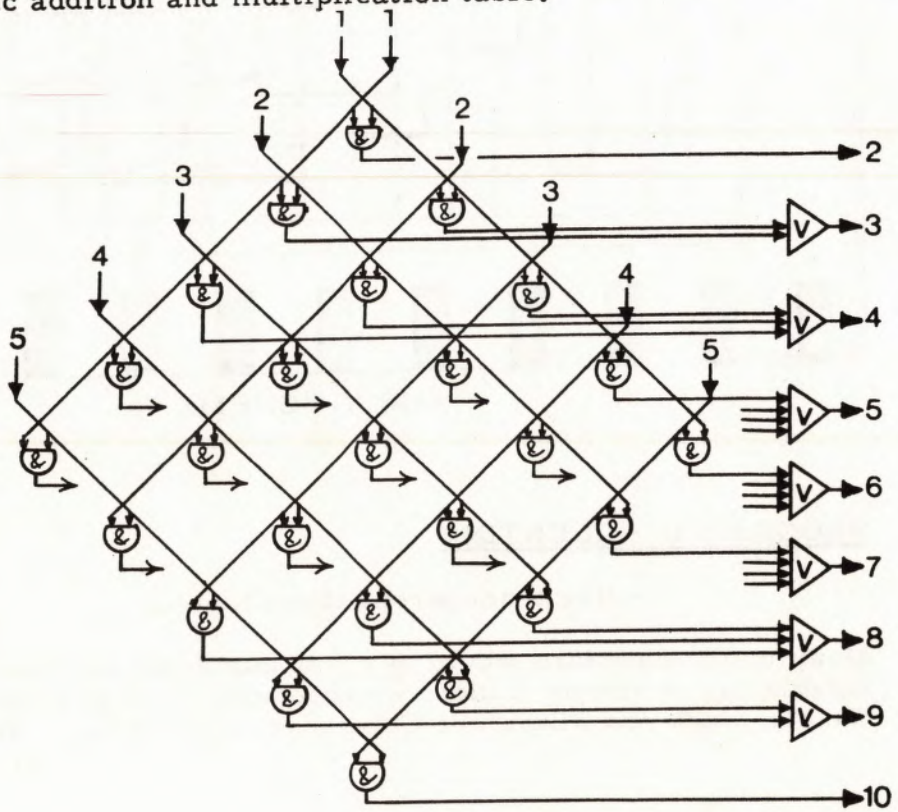


You can see how this improved circuit gets rid of the false answers that could occur in the earlier scheme.

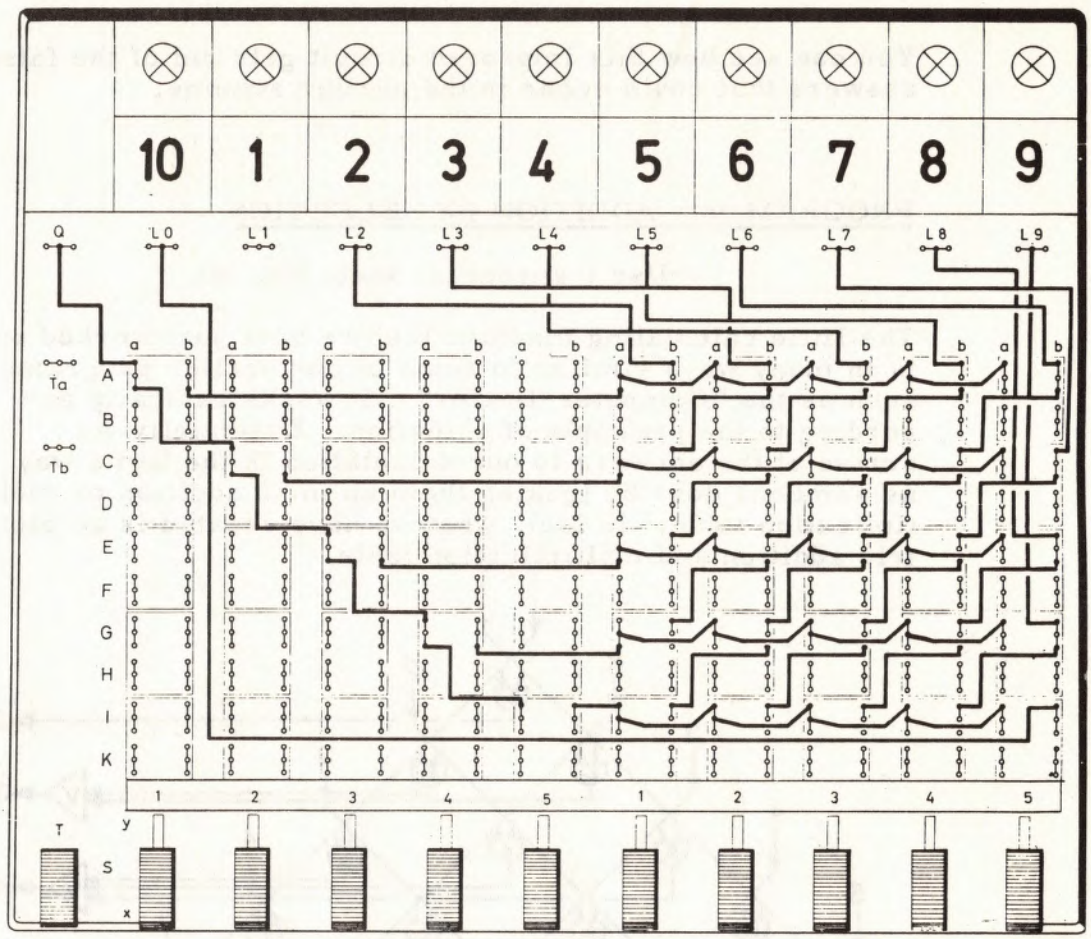
PROGRAM 30 - ADDITION BY SELECTION

-Use transparent sheet No. 30.

The little calculating machine that we have just worked out is in many ways similar to some of our earlier programs such as the "Computer-Doctor". It works basically according to the principle of selection. Essentially we arrive at the answers to our calculation in the same way as someone does by looking them up in an addition or multiplication table. In fact, what we have created is an electric addition and multiplication table.



Following this method, we can construct a calculating apparatus which allows us to add more than 2 and 3. The program, as the Logical Scheme shows, has nothing but AND circuits which are connected by OR circuits. This program allows us to add up to 5 and 5. You add a figure on the left to a figure on the right, and the result appears in the Output Section. Since our calculator finds the answers by going through a table instantly, we do not really have to rewire it to change our adding machine into a multiplying machine. To multiply, we only have to change the indicator chart for the answers, and it will work just as well. In this way, it will be easy to adjust this to perform other operations. We could create an indicator for the square or cube roots of numbers. We could even create new tables of multiplication, which might not have any use for us but would always give consistent answers.



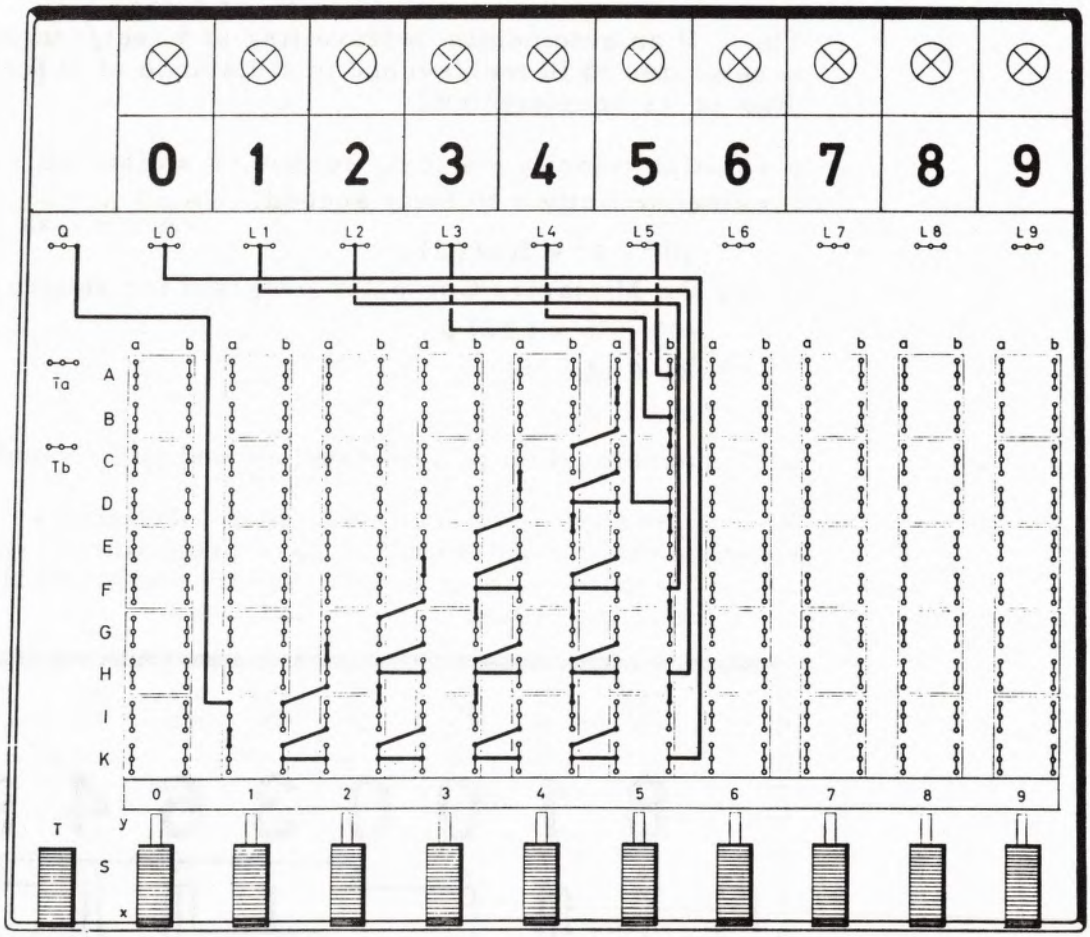
WIRING SCHEME 30

PROGRAM 31 - COUNTING

- Use transparent sheet No. 31.

Many mathematicians would say that our usual method of calculating is simply a more complicated form of counting. We can teach our Miniature Computer to count too. This

program shows you how to do this. It works like this: you can move any number of Sliders Nos. 1 - 5 to y. It makes absolutely no difference whether the three sliders are the first three, the last three, or the first, second and fifth. The computer will simply tell you the number of sliders you have moved. The Wiring Scheme, as you can see, is basically simple. It consists only of AND and OR circuits. Try it. Use any number of sliders, but remember only Sliders one to five are programmed.



WIRING SCHEME 31

PROGRAM 32 - MATHEMATICAL SQUARES

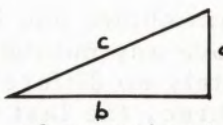
- Use transparent sheet No. 32.

A mathematical operation which is frequently used in high schools and colleges is the operation of squaring. To square a number is to multiply it by itself. This operation is indicated by placing a little number 2 above and to the right of the given number.

Thus: $5^2 = 5 \times 5 = 25$

This operation is useful in geometry in computing the area of a square shape. It is also useful in relating the lengths of the sides of a right-angled triangle - according to the Phythagoras Theorem: $a^2 + b^2 = c^2$

For example: $3^2 + 4^2 = 5^2$



In physics too, the operation of squaring is very popular - as for example in the following formula based on Newton's laws of motion. This formula relates the final velocity (v) of a body under acceleration to its initial velocity (u), the acceleration (a) and the distance the distance travelled in the interval (s) .

$$v^2 - u^2 = 2as$$

Thus, if an automobile is travelling at 4 feet/second and accelerates to 10 feet/second in a distance of 21 feet..... what is its acceleration?

u = initial velocity = 4 feet/ second, s = distance = 21 feet
v = final velocity = 10 feet/ second.

$$v^2 - u^2 = 2as$$

$$10^2 - 4^2 = 2(a)(21)$$

Using the Miniature Computer program for squares we get:

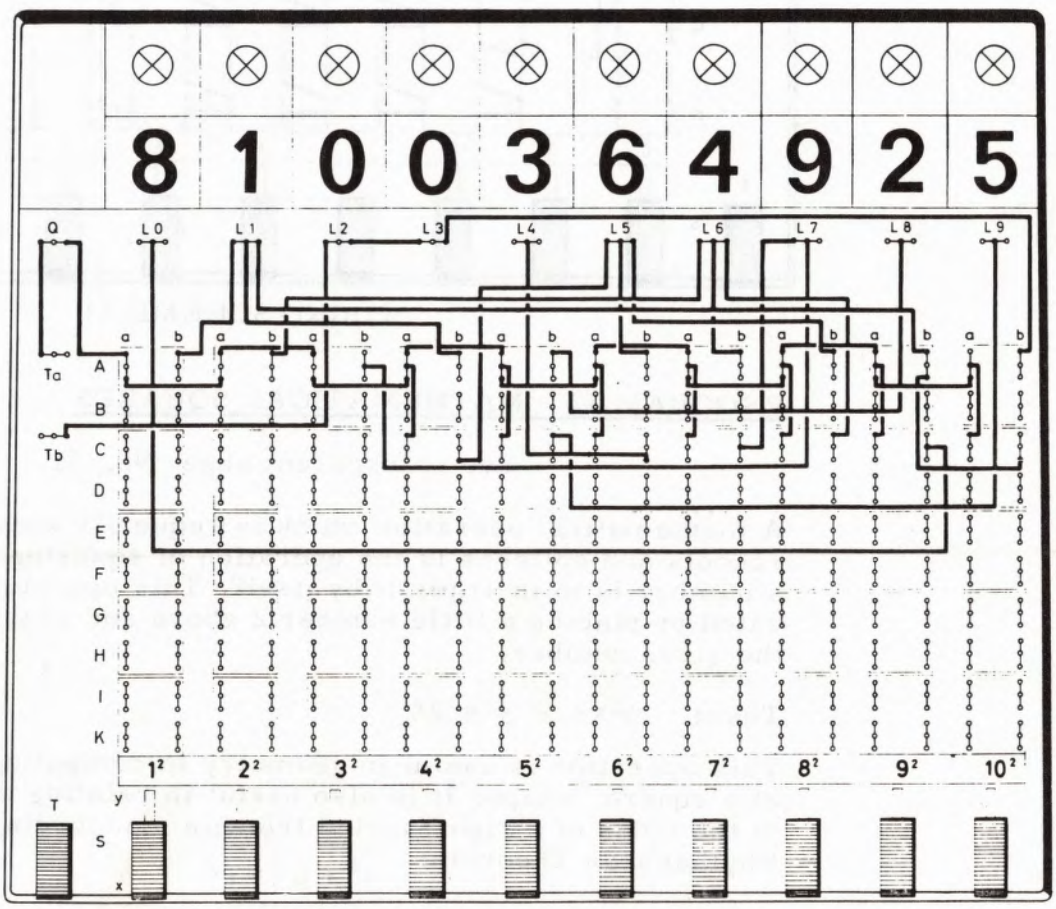
$$100 - 16 = (42)a$$

$$84 = 42a$$

$$a = 2 \text{ ft/sec}^2$$

or the acceleration is 2 feet per second per second.

You can probably think of other uses for squares - but to simplify the computation, we have programmed your Miniature Computer to give you the squares of all whole numbers from one to ten.



WIRING SCHEME 32

PROGRAM 33 - THE BINARY SYSTEM

- Use transparent sheet No. 33.

We have already mentioned the Binary System and told you something about it. In Part II, this system is able to show numbers as electrical circuits which have only two states: In practice we represent the two states of an electric circuit, OFF and ON, by the symbols O and L. In this system, O means OFF and L means ON. O means no current and L means current.

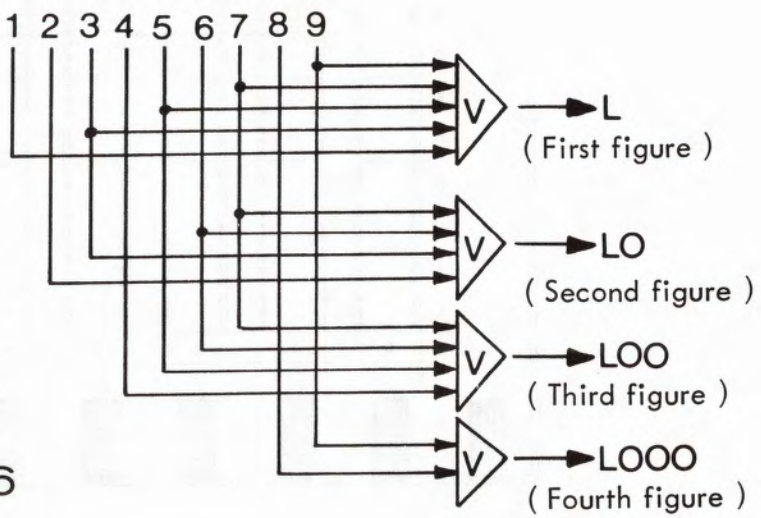
In your Miniature Computer, L represents one of the bulbs lit up and O represents a bulb that is out. Later on we will be using these symbols in binary calculation.

Our visual system of counting is based on the number 10. For this reason, it is known as the decimal system. We count from 1 to 9 and then comes 10. The rest of the numbers are all made up of previous digits. We multiply 10 x 10 and we get three digits. Mathematicians would say that our system is based on the powers of 10.

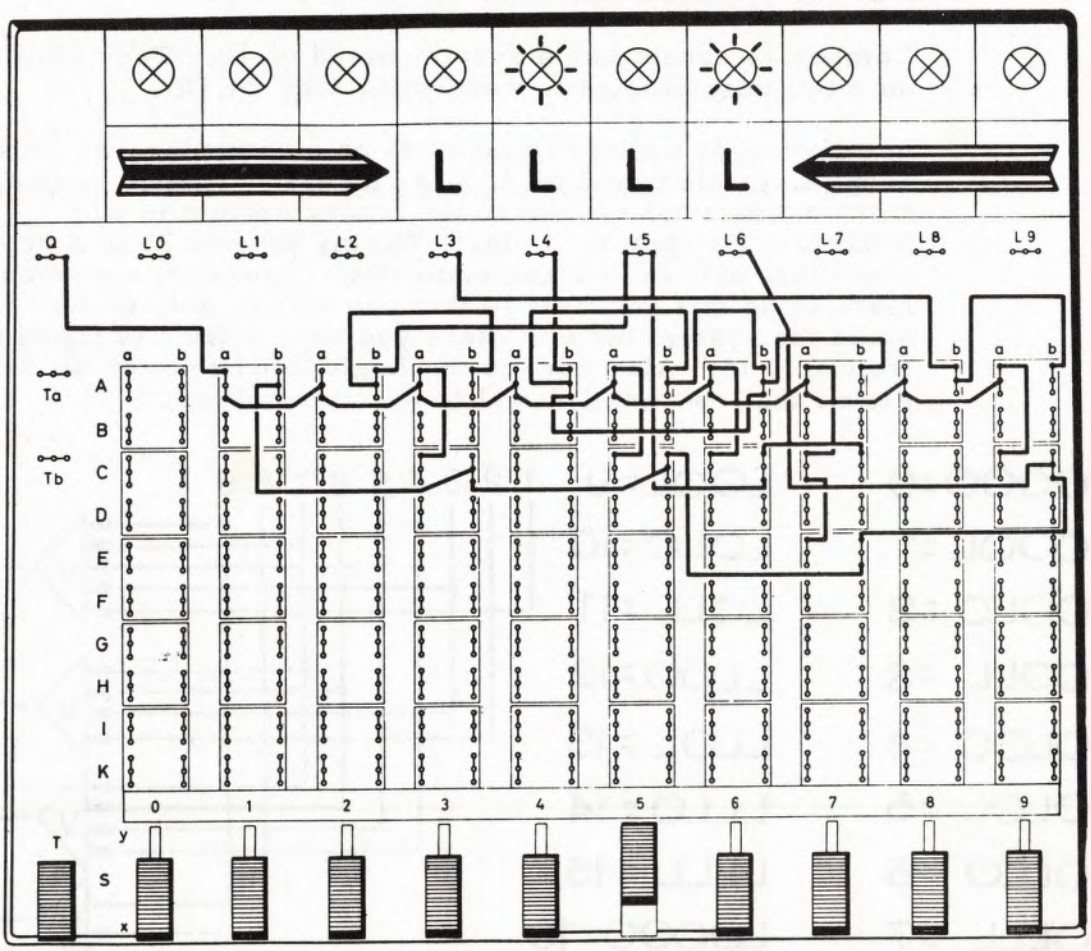
Computers cannot use a system based on 10. They have only two possibilities to work with: OFF or ON.

In order to do mathematical work on a computer, we must adopt a system based on 2. We can easily change numbers from the decimal system to the binary system in our Miniature Computer. In fact, Wiring Scheme 26 is a program that will do this automatically. However, you should learn to do this yourself so that you will be able to understand the system better. Here you have a table of figures from 0 to 16. Also here is the Logical Scheme for all programs that convert decimal to binary numbers.

OOOO=0
OOOL=1
OOLO=2
OOLL=3
OLOO=4
OLOL=5
OLLO=6
OLLL=7
LOOO=8
LOOL=9
LOLO=10
LOLL=11
LLOO=12
LLOL=13
LLLO=14
LLLL=15
LOOOO=16



You interpret this program in the following way: when a bulb lights up, it signifies an L. If the bulb stays off, it signifies an O. As in the decimal system, the binary system of numbers is a place system. This means that the size of a number depends on its position. Just as in the decimal system, 10 = 1 x 10 and 100 = 10 x 10 and 1000 = 10 x 10 x 10, and so forth; thus the value of the numbers in the binary system is also changed by how many places to the left each figure is. But instead of increasing in multiples of 10 as in the decimal system, in the binary system the numbers increase by multiples of 2, as the digits move to the left. In both systems the figure furthest on the right represents 1. The next place from the right in the binary system represents 2. The third one over represents 4. The fourth one from the right is 8. Next comes 16, 32, 64, etc. You can see that by using O's and L's alone, any number that can be shown in the decimal system can easily be shown in the binary system. It may take more digits to show them in the binary system; but because it is made up of only two types of digits, it is perfectly suited for use in electronic calculators.



WIRING SCHEME 33

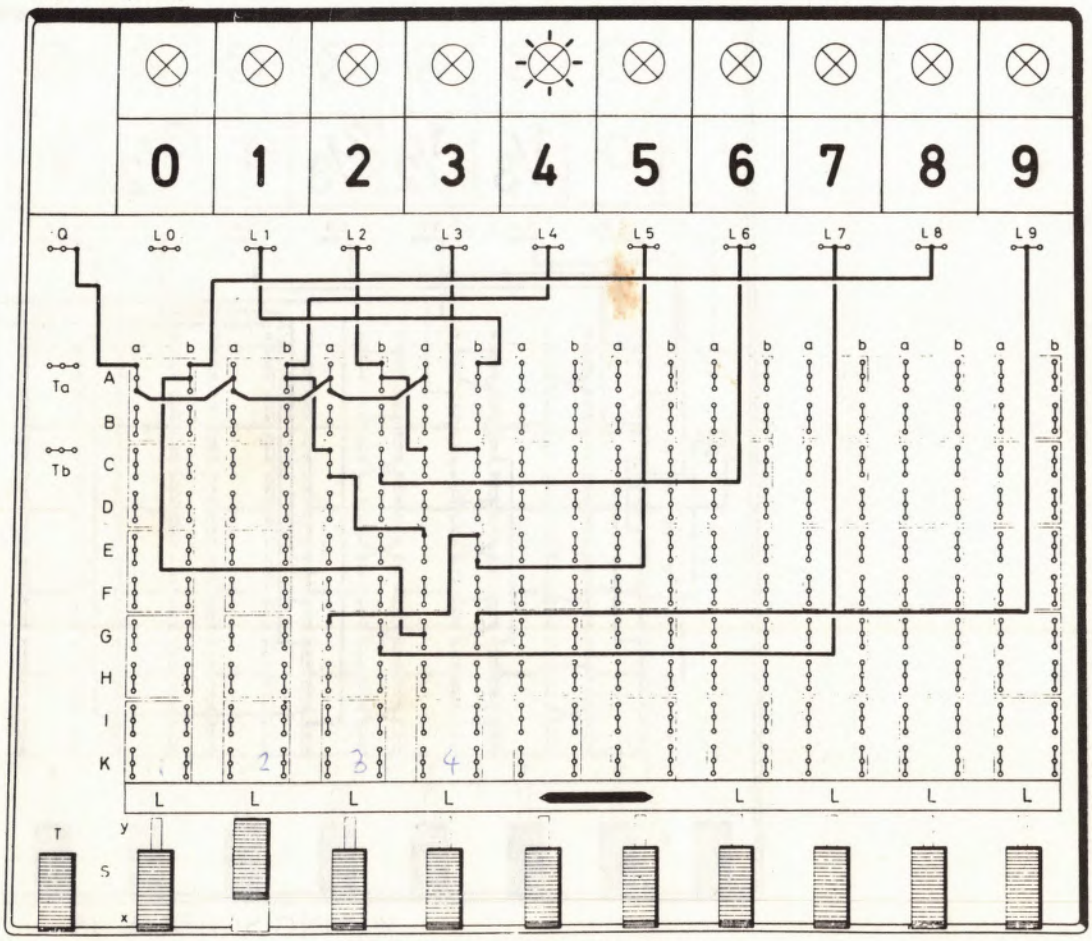
PROGRAM 34 - BINARY TO DECIMAL CONVERSION

- Use transparent sheet No. 34.

Here is a new program that does exactly the opposite of what we did in the last one. In this program, we change binary figures into decimal figures. This program has one strange thing in it: you will sometimes see that several bulbs light up at the same time in the Output Section. This should not bother you. In this program, the figure that lights up furthest to the right (that is, the one that has the highest numerical value), is the correct answer.

Just because several bulbs light up at the same time does not mean that there is a mistake in your wiring. Let us look more closely at it. For example, take the binary figure OLOL which signifies 5. Here we also see 4 (OLOO) and 1 (OOOL) light up. If you look at this a little more closely, you will notice that the figures shown always have a certain relationship with each other. That is, these figures always add up to the number that is our answer.

We can make this a better program and get rid of these extra results by introducing NOT circuits in the form of SINGLE-PATH circuits, as in Wiring Scheme 29b. By introducing these elements, you can produce a wiring scheme that will show only the one answer you are looking for. We leave that up to you.



WIRING SCHEME 34

REAL COMPUTERS WORK THIS WAY TOO!

Large electronic calculators meet the same problems when changing numbers from the decimal to the binary system and vice versa. To do this, they are programmed with number-translators which follow programs that look like the one our Miniature Computer has just finished. It is interesting to see that we can also code letters and words as binary numbers. This process is used in telegraphic printing machines, such as Telex and TWX. Perhaps you have seen a perforated band which these machines sometimes use. Each hole represents an L and the lack of a hole represents an O.

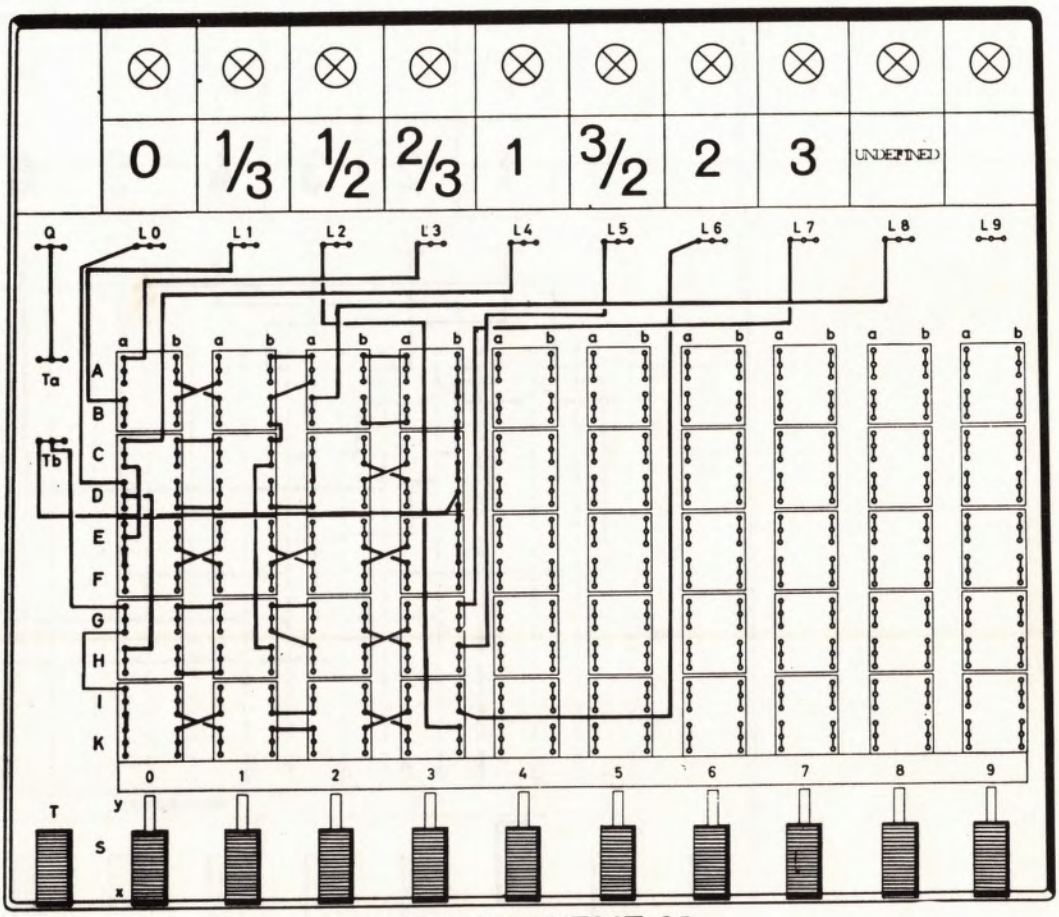
PROGRAM 35 - DIVISION PROGRAM

by David and Steven Prince of Bridgewater, N.S.

- Use transparent sheet No. 35.

As another example of binary input and decimal output, you may try this division program.

Sliders 0 and 1 are used to enter the dividend (numerator), and Sliders 2 and 3 to enter the divisor (denominator). Since



WIRING SCHEME 35

only two sliders are available for each number input, and since binary input is used, we can enter the following possible numbers: 0, 1, 2, 3.

In mathematics, division by zero is not permitted, so if you enter a zero for the divisor, the "Undefined" should light up on the panel. On the other hand, if a zero is entered for the dividend, the result will be zero and that light will shine on the panel.

Now consider the number 0/0. Should this be zero because a zero is in the numerator? Should it be undefined because a zero is in the denominator? Or should this equal "one" because we have the same number in the numerator as in the denominator? According to the laws of mathematics, the correct answer must be "undefined".

PROGRAM 36 - HOW TO CALCULATE WITH O AND L

- Use transparent sheet No. 36.

How does one actually use these binary figures to calculate? It is really quite simple. Let us take the sum of 3 and 4. According to our table, 3 and 4 are in binary terms:

```
  OOLL
+ OLOO
------
  OLLL
```

Let us check this answer. OLLL is 7. The calculation is correct. Let us try another, 2 + 4.

```
  OOLO
+ OLOO
------
  OLLO
```

Is this correct? Yes, OLLO is 6. Let us do more calculations. What happens with 2 + 6?

```
  OOLO
+ OLLO
------
 OL2LO
```

This sum does not fit well into our system, since we only know O's and L's. OL2LO does not exist. Is our system of calculation at fault? No, because we have to learn another small rule of addition. Ordinarily, in our usual system of numbers we would say that 1 + 1 = 2. It is different in the binary system. In this system L + L = LO.

This seems completely ridiculous, but actually in the binary system it is really true: L + L = LO. In practice, you write O and carry L.

Using this new knowledge let us again try to add, properly

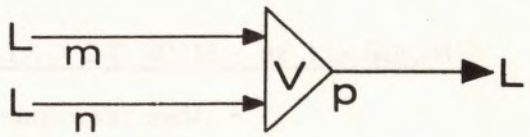
this time:

```
  OOLO
+ OLLO
------
  LOOO
```

This answer is correct.

Now that we can do it in theory, let us see if we can use this method of addition with our Miniature Computer. First let us review the rules of addition in the binary system which we have just learned.

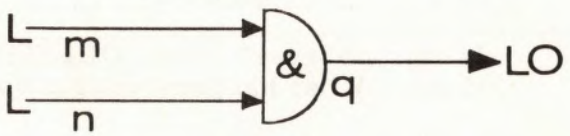
Here are three simple formulas: O + O = O, O + L = L, L + L = LO. These rules of addition are very easy to show on your Miniature Computer. The logical steps are very simple. For O + O = O and O + L = L, we have the Logical Scheme:

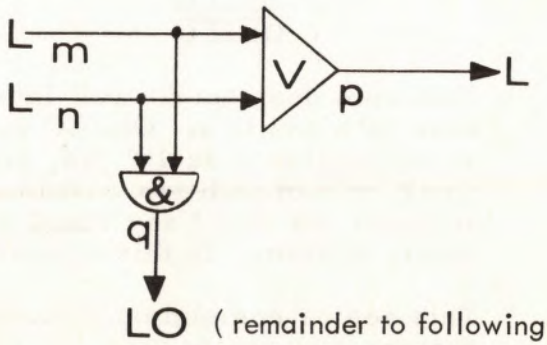


If an L (an electrical impulse) is produced on wire m or n, either of these will be carried by the OR circuit and they will have the result L. This gives us the result O + L = L or L + O = L.

At the same time, if no current is produced in either m or n, there will be no current in circuit p. This means that O + O = O. This is also true.

However, now we have to show that L + L = LO. This is quite simple. We will use an AND circuit for this. All we have to do is to bring the two circuits together in order to have all binary operations in this Logical Scheme:



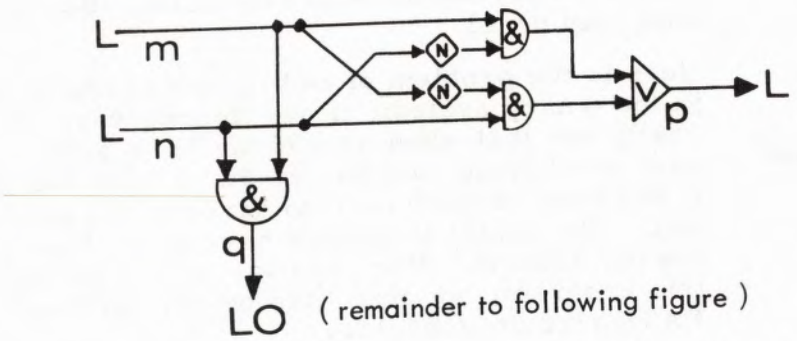


(remainder to following figure)

But is it true? No. It is not completely right, because although the answer gives the needed LO, the OR circuit will still work as well and the result we will get, will be two L's (LL).

Our only other choice is to use two NOT circuits in the form of SINGLE-PATH circuit.

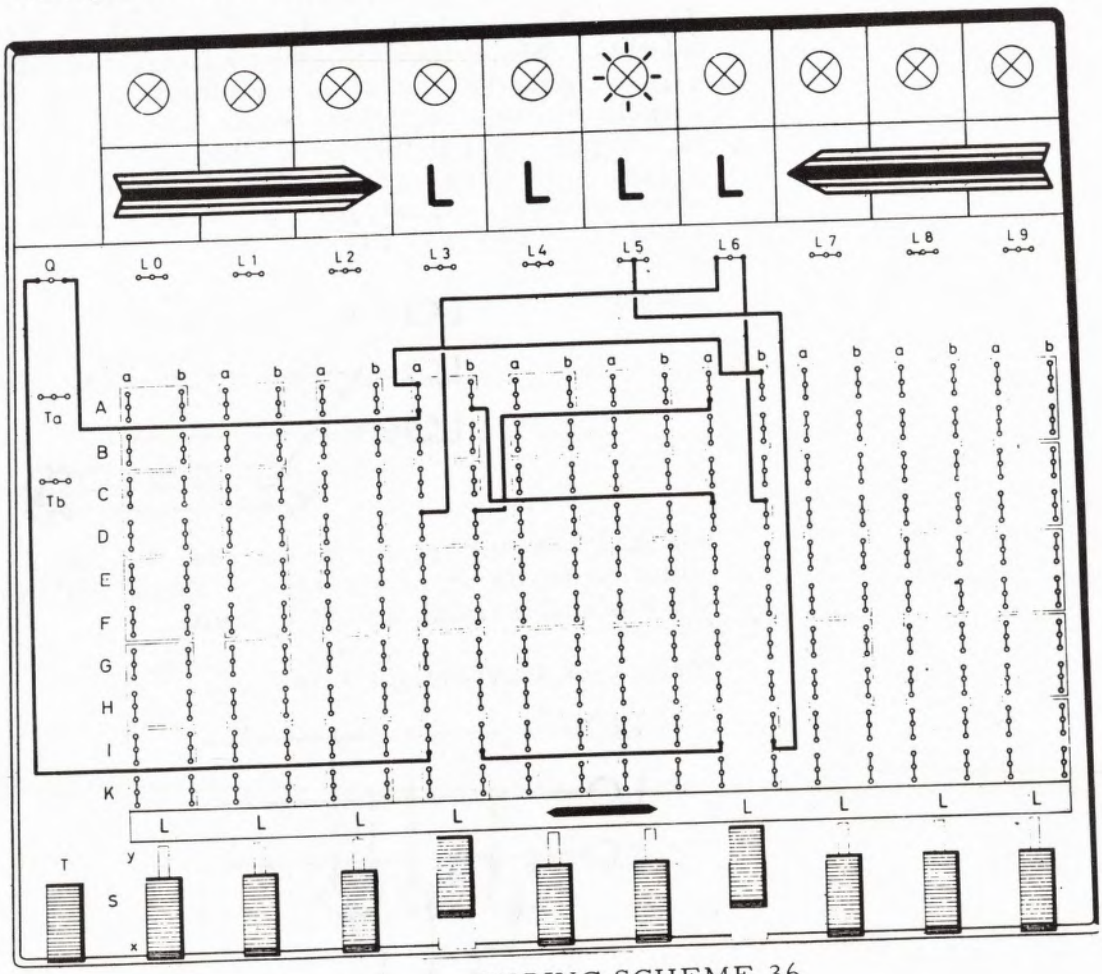
The result will be this:



(remainder to following figure)

Now all we have to do to have the perfect scheme for the addition of two binary figures, is to add two sliders to m and n and two bulbs to p and q.

In Wiring Scheme 36, we show the actual wiring for such a binary adder for two figures.



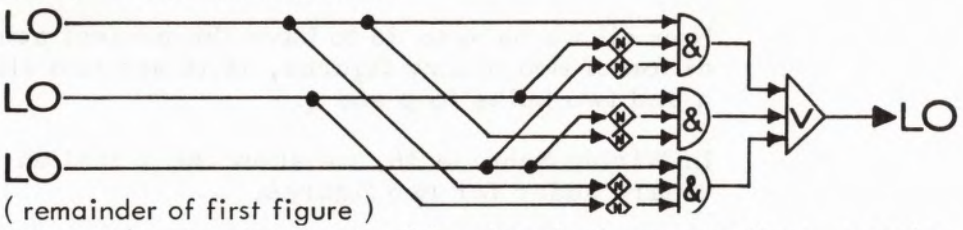
WIRING SCHEME 36

PROGRAM 37 - A TWO-PLACE BINARY ADDER

Now we can enlarge Program 36 so that it will be able to handle binary numbers of two digits. But this is easier said than done!

```
 +LL
  LL
 ----
  L    carried from column on right
 ----
 LLO
```

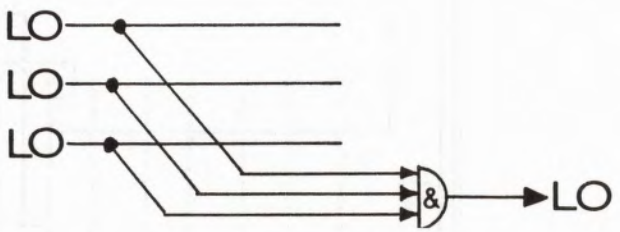
Here is the problem of adding two numbers of two digits in the binary system: if our figures were LL + LL we could easily see that when we add the 'L's on the right side of each number our answer would be LO. We put down the O and then we have an L to carry to the next column to the left. The result is that we have three figures to add for the left column. Our program must consider all this. In this program, we must also be careful to add SINGLE-PATH circuits like this:



(remainder of first figure)

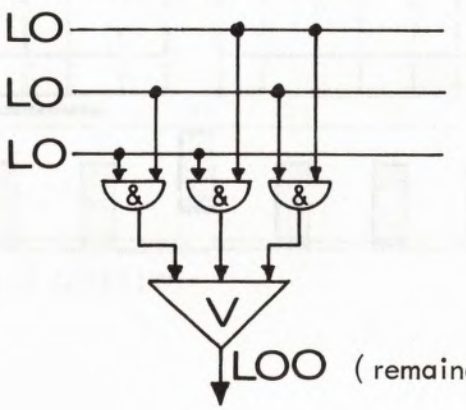
This is the problem: in this circuit there are three possibilities. Since this situation exists, the SINGLE-PATH circuit must get rid of everything that can happen when two impulses come together, since L + L = O, with a remainder of course. But if the three conductors all receive an impulse, it is necessary to show L since L + L + L = L (with a remainder). Therefore we must provide an answer which joins the three circuits.

This is it:



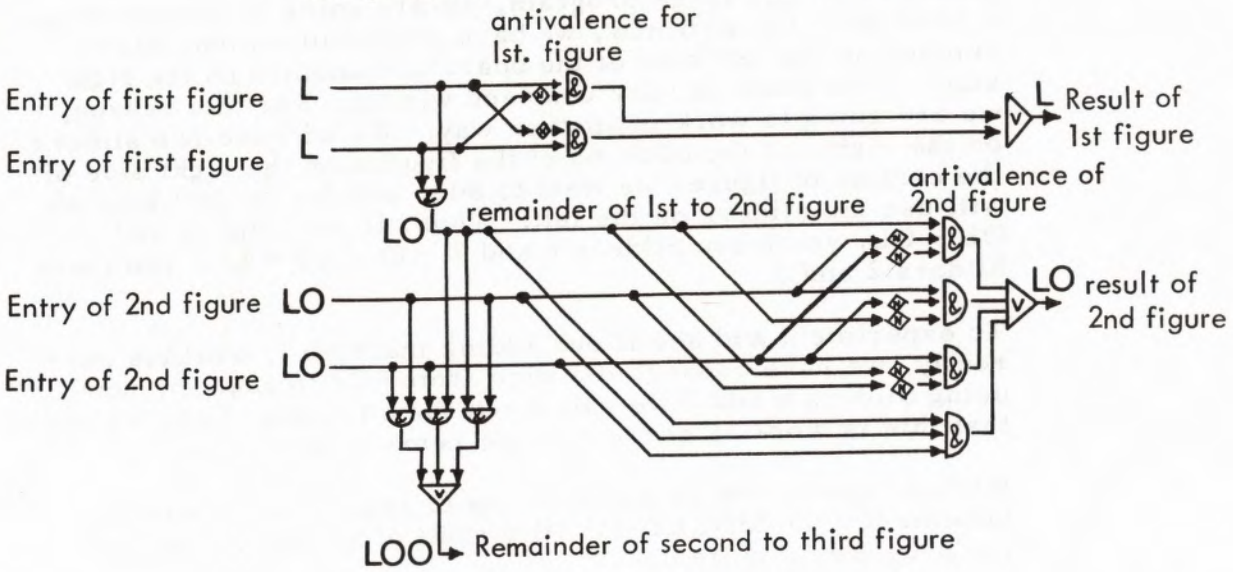
An AND circuit for the remainder is also necessary. It must be made up of a series of three AND circuits joined to each other through an OR circuit.

Here is a diagram for this:



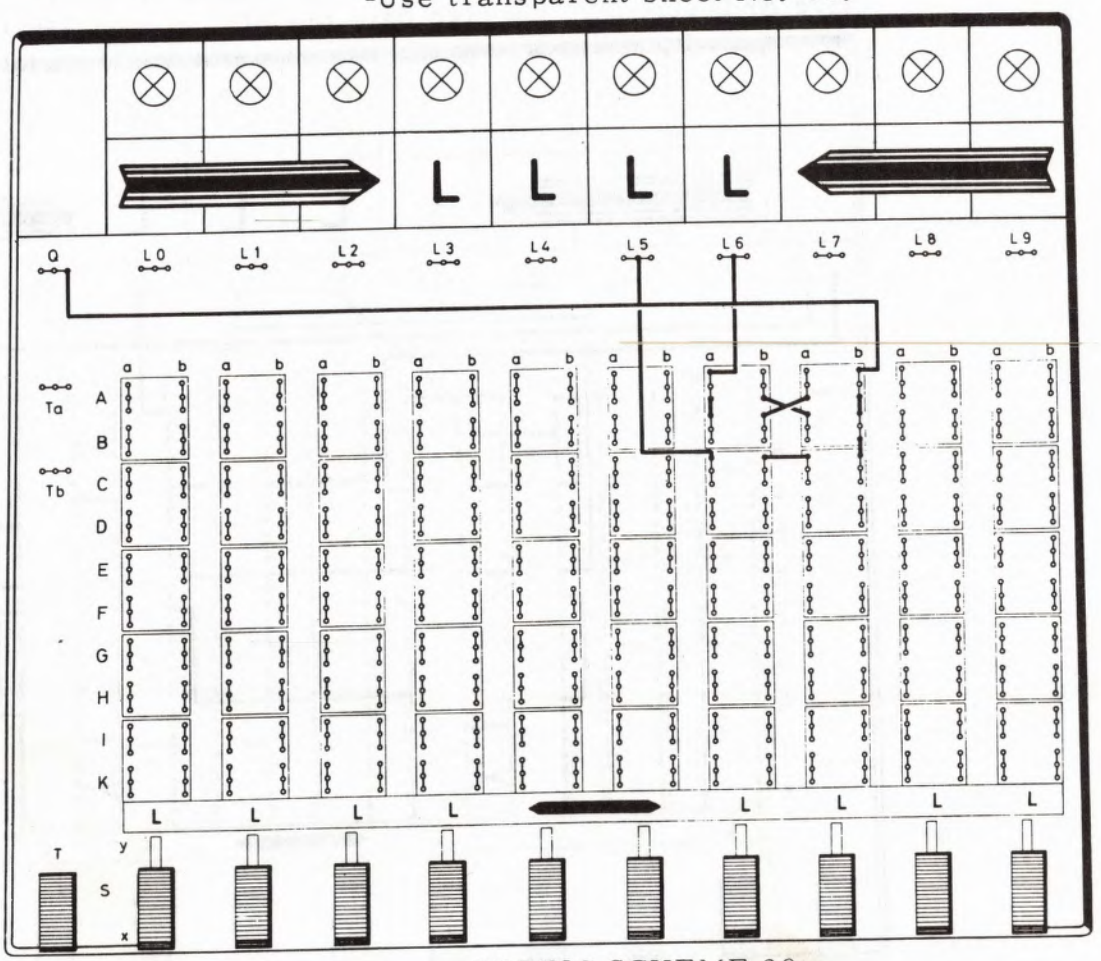
(remainder of 2nd to 3rd figure)

If you have understood all of this, you can combine this partial program into our giant wiring diagram.



PROGRAM 38 - THE GIANT PROGRAM FOR ADDITION IN THE BINARY SYSTEM

-Use transparent sheet No. 38.



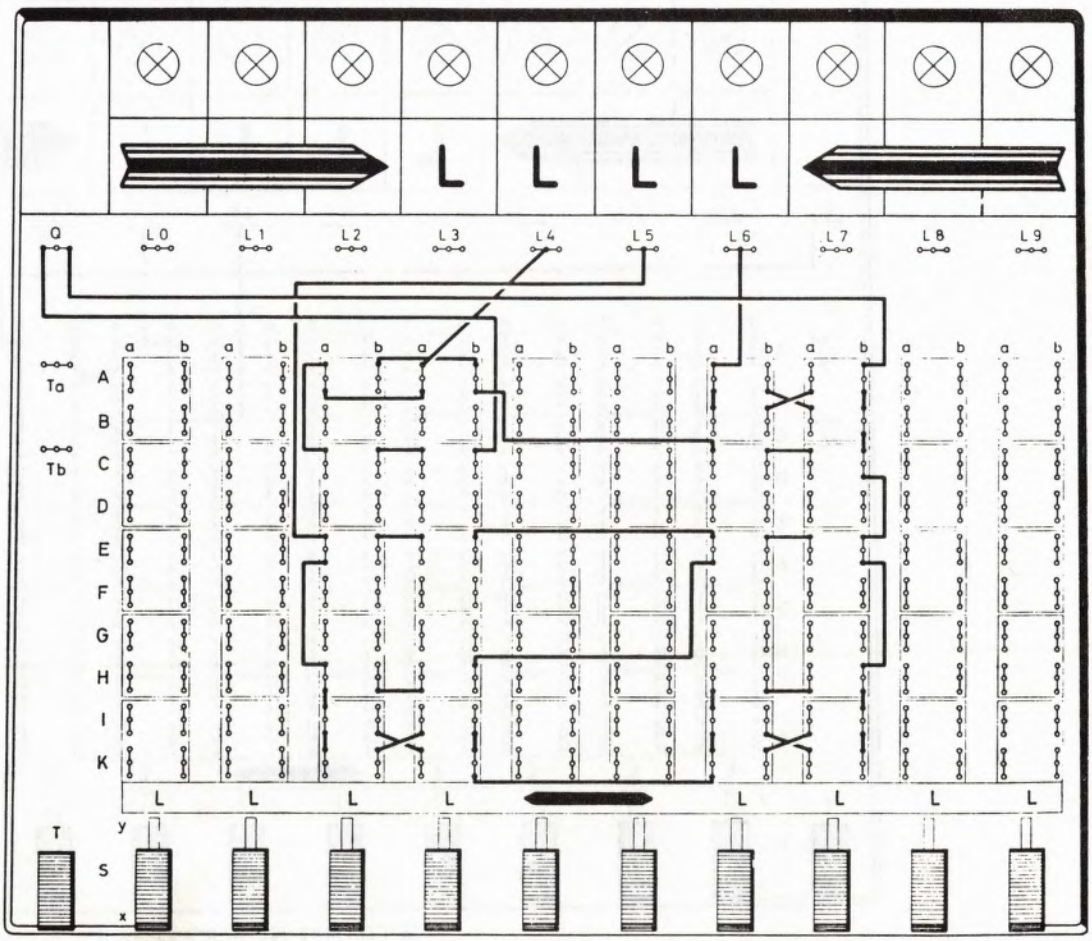
WIRING SCHEME 38a

Once you have understood the logic of this, it is quite easy to apply it to the Programming Board of your Miniature Computer. For this large program, we are going to change things a little bit. Up until now, we have programmed one binary number on the left side of the board and another on the right side. This leads to very confused wiring. For this reason, we are going to work in another way. We will use two sliders on the right for the addition of the figures on the right side of the column of figures we want to add, and for the left side we will use the sliders on the left. Thus, if you want to add OL + OL, you move Sliders 6 and 7. For LO + LO, you move Sliders 2 and 3.

To experiment and see if our adding machine is working correctly, we have connected a wire from 6Ca to Light 5; now, using Sliders 6 and 7 we can check the addition. Once we see that this is working correctly, we remove this wire.

Wiring Scheme 38b is added on top of 38a, it is the wiring scheme for the SINGLE-PATH circuit of the second part of the programming field.

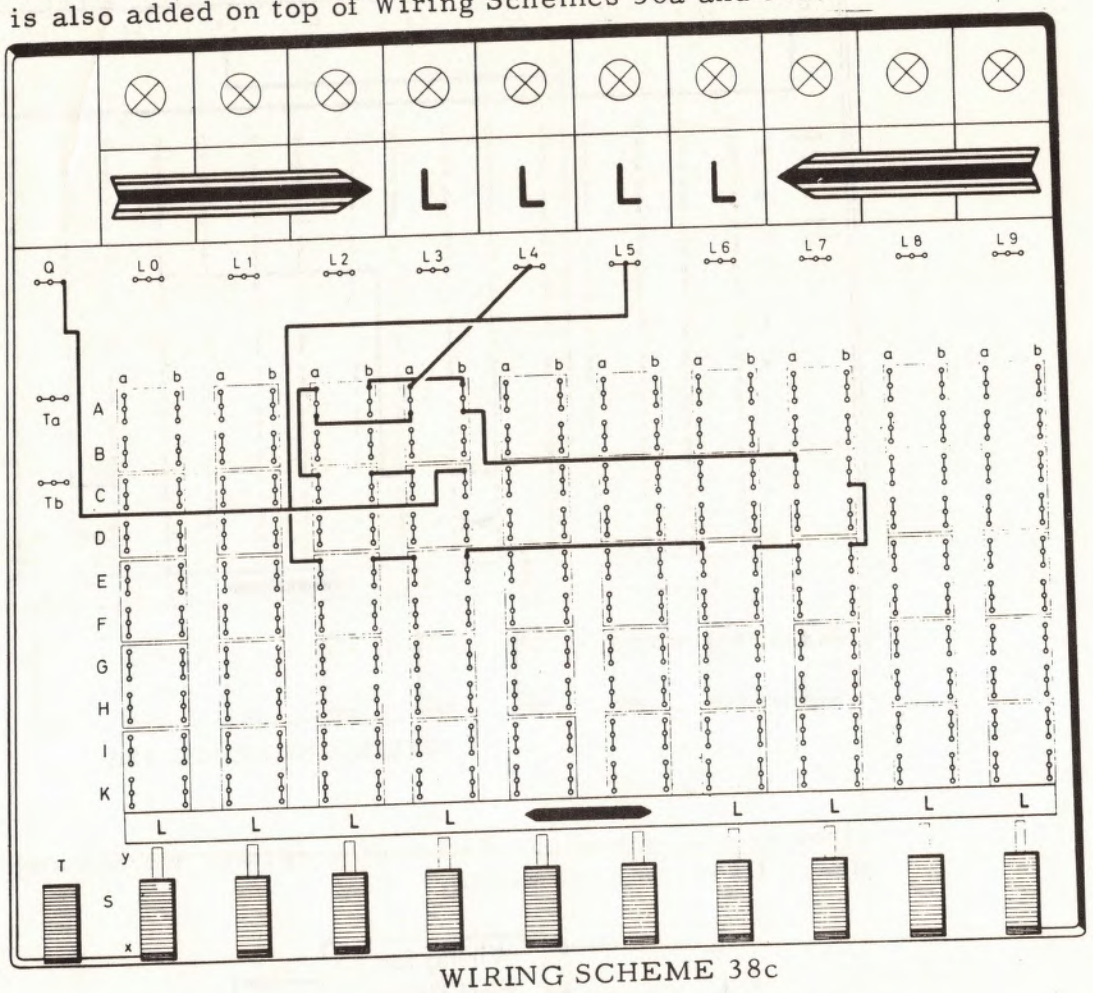
At first, it seems that there are only two SINGLE-PATH circuits, even though in the Logical Scheme there are three. However, you should also note that we have NOT circuits joined by an AND circuit at 3Hb/a to 2Hb/a, which is the connection on top with the branching elements 6C to 7C in a positive AND circuit. We also have the NOT circuit 7Hb/a to



WIRING SCHEME 38b

6Hb/a. It is interesting to think how confused the working of this calculator would be without these SINGLE-PATH circuits.

What is missing now are AND circuits for the remainder of the second column to be added to the third column. You will find these in Wiring Scheme 38c in the left top. This is also added on top of Wiring Schemes 38a and 38b.



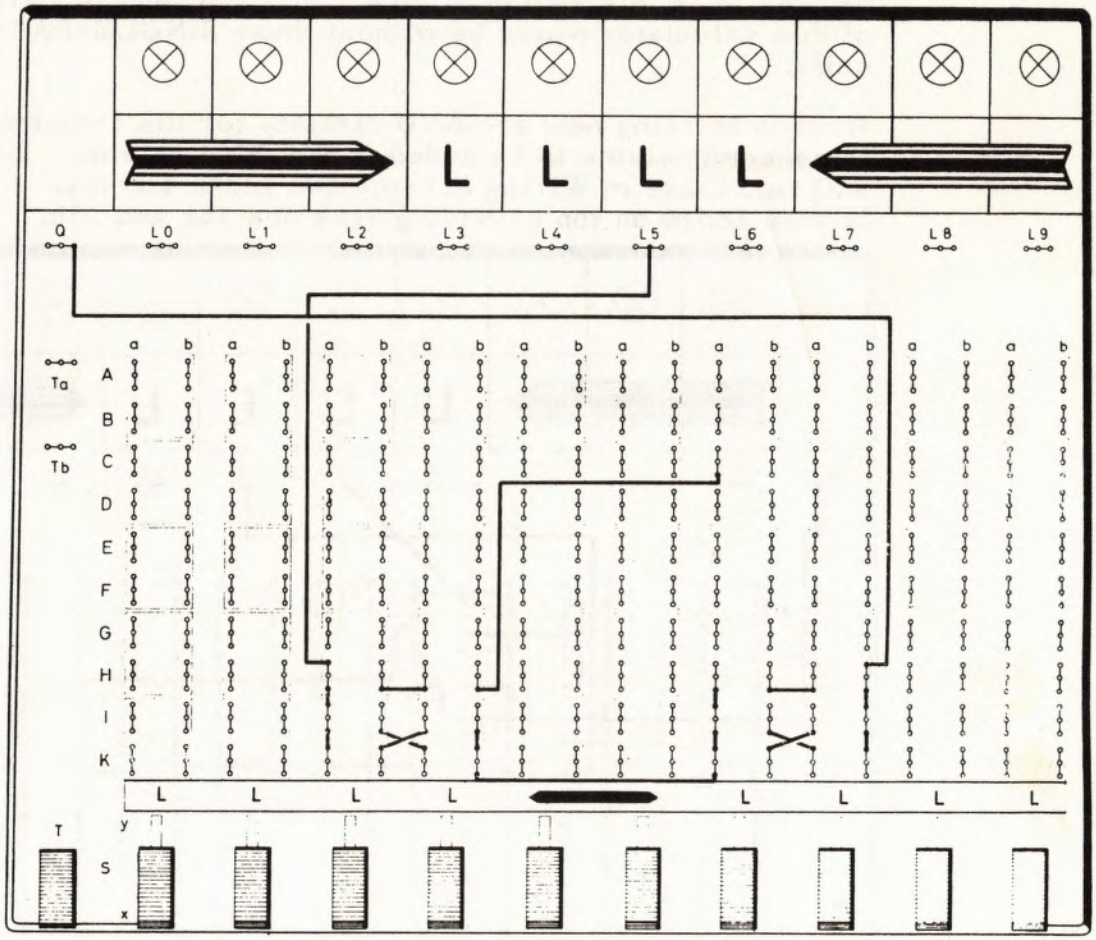
WIRING SCHEME 38c

Now it is clear that changing from Scheme 38a to 38c gives us Scheme 38d. If you look at this closely, you will see that several wires are connected differently. This has been done for certain electrical reasons and not for logical or mathematical ones. This gets rid of the problem of lighting certain false bulbs and of the current passing through certain wrong paths. We have also simplified the connection to Q. Our calculating program (38d) is thus finally finished!

REAL COMPUTERS WORK THIS WAY TOO!

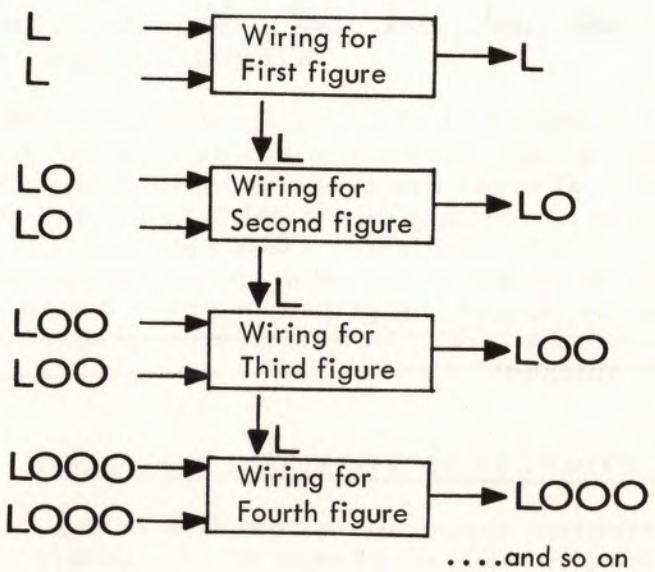
This program seems more complicated than it really is. It is actually made up of very simple unitary circuits. The only problem is that we need a great many of them.

In Scheme 38d, we have gathered together all the partial



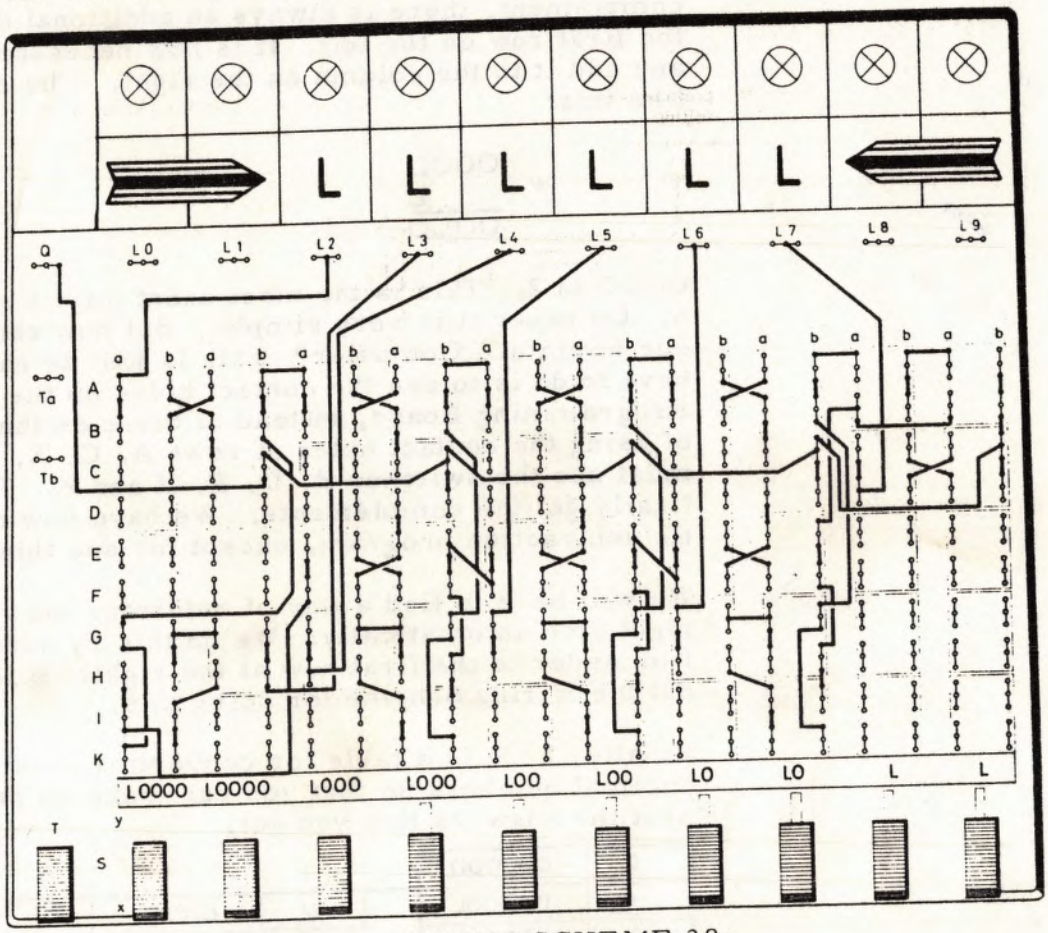
WIRING SCHEME 38d

wires of 38a, b and c. With this program, we can add binary figures of two digits.



You will now ask why we can't add more figures. It is only because of the lack of space on the Programming Board that our Miniature Computer cannot add infinitely

large numbers. With unlimited programming space we could theoretically go on infinitely, as in the Logical Scheme. This is actually done in large computers. With the greatest economy of design, we can make up a program for two binary numbers of five figures as in Scheme 38e.



WIRING SCHEME 38e

PROGRAM 39 - SUBTRACTION IN THE BINARY SYSTEM

Large computers do not really multiply or divide. What they really do is repeat additions or subtractions until they get the correct answer.

But how do we subtract in the binary system with our Miniature Computer? It is really quite simple. We do it by adding the complement.

What is the complement? It is simply the opposite. The complement of L is O and that of O is L. The complement of the figure LOOL is OLLO and that of 4 would be LOLL. To subtract 4 from 6, the calculation would go as follows:

$$\begin{array}{r} 6 \\ -\ 4 \\ \hline \end{array} \rightarrow \begin{array}{r} \text{OLLO} \\ -\ \text{OLOO} \\ \hline \end{array} \rightarrow \begin{array}{r} \text{OLLO} \\ +\ \text{LOLL} \\ \hline \text{LOOOL} \end{array}$$

What does LOOOL represent? This cannot be right. 6 - 4 is not 17, even when calculating with a computer. We must have made a mistake somewhere.

Not at all! This is binary arithmetic. When adding the complement, there is always an additional digit, the L of the first row on the left. It is now necessary to go back and add it to the column on the right. The result looks like this:

$$\begin{array}{rr} & \text{OOOL} \\ + & \text{L} \\ \hline & \text{OOLO} \end{array}$$

OOLO is 2. This is the most exact way to subtract 4 from 6. On paper it is very simple. But how can we do it with our Miniature Computer? This is just as easy! All we have to do is to use the contact holes on the bottom of the Programming Board, instead of those on the top. Instead of using the contact holes of rows A, C, E, G and I; we must use the switches B, D, F, H and K. Thus we automatically get the complements. We have now made an automatic subtraction program, except for one thing.

We will have to find a way of returning the remainder to the right side automatically. We do this by adding the L of the remainder to the first row at the right. It is up to you to work out the Wiring Scheme for doing this.

Finally, here is a table for converting binary numbers into decimal numbers so that you can make up problems and interpret the answers that you get.

0	OOOOOO						
1	OOOOOL		22	OLOLLO		43	LOLOLL
2	OOOOLO		23	OLOLLL		44	LOLLOO
3	OOOOLL		24	OLLOOO		45	LOLLOL
4	OOOLOO		25	OLLOOL		46	LOLLLO
5	OOOLOL		26	OLLOLO		47	LOLLLL
6	OOOLLO		27	OLLOLL		48	LLOOOO
7	OOOLLL		28	OLLLOO		49	LLOOOL
8	OOLOOO		29	OLLLOL		50	LLOOLO
9	OOLOOL		30	OLLLLO		51	LLOOLL
10	OOLOLO		31	OLLLLL		52	LLOLOO
11	OOLOLL		32	LOOOOO		53	LLOLOL
12	OOLLOO		33	LOOOOL		54	LLOLLO
13	OOLLOL		34	LOOOLO		55	LLOLLL
14	OOLLLO		35	LOOOLL		56	LLLOOO
15	OOLLLL		36	LOOLOO		57	LLLOOL
16	OLOOOO		37	LOOLOL		58	LLLOLO
17	OLOOOL		38	LOOLLO		59	LLLOLL
18	OLOOLO		39	LOOLLL		60	LLLLOO
19	OLOOLL		40	LOLOOO		61	LLLLOL
20	OLOLOO		41	LOLOOL		62	LLLLLO
21	OLOLOL		42	LOLOLO		63	LLLLLL

PROGRAM 40 - MINI-CHECKERS

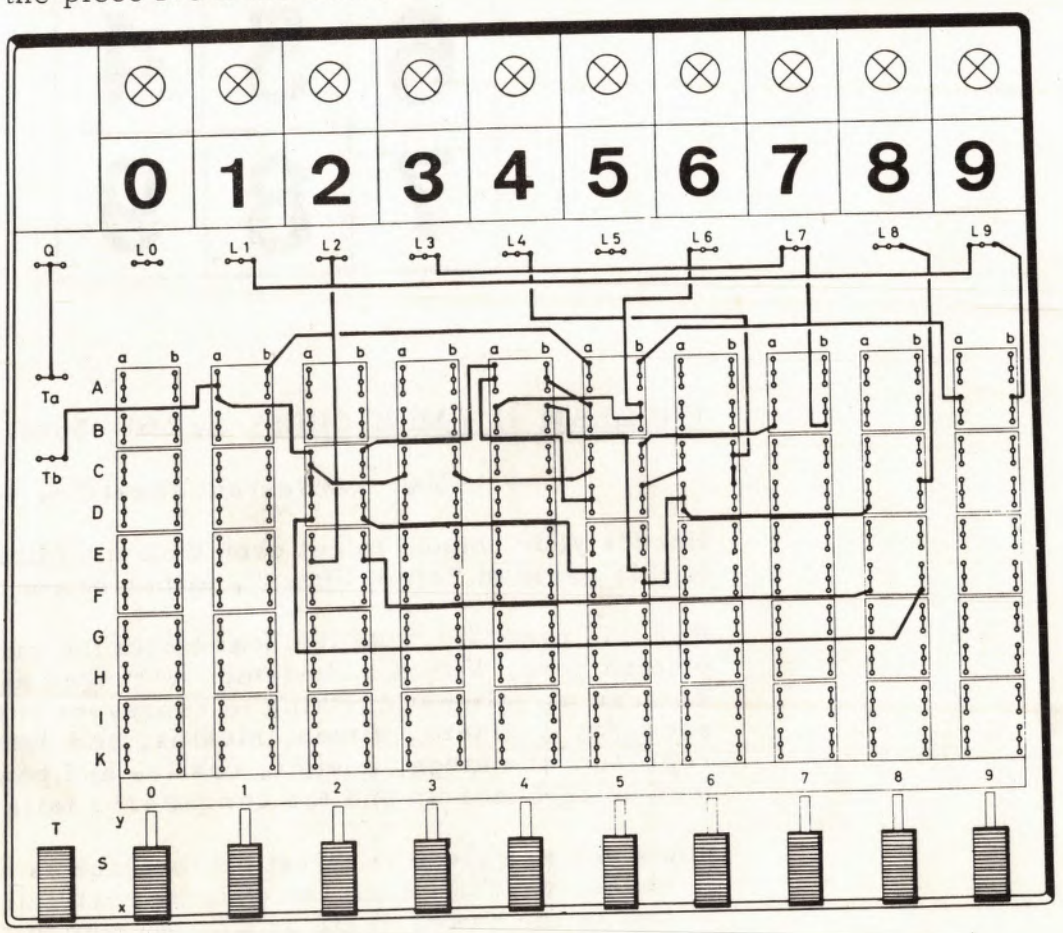
-Use transparent sheet No. 31.

Have your friends challenge your Miniature Computer to a game of mini-checkers. Your computer will win every time!

For this program, use the "board" with nine squares printed here, or draw your own on a separate piece of paper. You can use almost anything to make checker pieces. Six coins would work well; for instance your men can be pennies while the computer uses nickles.

You and the computer each start with three pieces. First place your three checkers on squares 7, 8 and 9 and place the Miniature Computer's pieces on squares 1, 2 and 3.

The rules are simple. You may move only in a diagonal and forward direction and only one step at a time. When it is your turn, and if one of your opponent's pieces is in the next square diagonally ahead of your piece, you may "jump" over him and "take him off". If you get a chance to take off one of your opponent's checkers, you must take it. You cannot give up a jump for "long-range strategic reasons". When one of your pieces has arrived at the last row, remove the piece from the board.



WIRING SCHEME 40

The goal of the game is to capture as many of the computer's pieces as possible. You must move first.

Example: When you start the game, the sliders for all squares that have checkers on them are set into the y position. Suppose your first move is from square 9 to square 5. You move Slider 9 to x and Slider 5 to y. Press the button and your Miniature Computer shows you its move by lighting up some of its bulbs: in this case, 1 and 9. Since your Miniature Computer has no arms, you must move for it; and the computer takes off the piece that you have just moved in your turn (the piece at square 5). As a result of this computer move, square 1 is empty (set Slider 1 to x); square 5 is empty (set Slider 5 to x); and the computer's checker has arrived at an end square, 9, which we now remove from the board (Slider 9 remains in the x position). It is now your turn to move.

When no more moves are possible, the game has ended. The winner is the one who has captured more of his opponent's pieces.

1	2	3
6	5	4
7	8	9

PROGRAM 41 - MINI-CHESS by Mike Sobel of St. Paul, Minn.

- Use transparent sheet No. 41.

Here's your chance to get even with the Miniature Computer. In this game of "Mini-Chess", either of you might win.

For this program, use the board with the sixteen squares printed here, with the chessmen arranged as in the diagram. You can use almost anything to represent chessmen; for example: quarters, dimes, nickles, and pennies may represent the kings, queens, castles and pawns with your men being heads up and the computer's tails up.

Since you are given the first move which is a big advantage in chess, we shall even the odds by restricting your starting moves to the pawns at B2 or C2. Furthermore, if you have an opportunity to capture more than one of the computer's

pieces, you must capture the highest possible piece (i.e. queen, castle, pawn - in that order). The pieces move like in the real game of chess, with the exception that the king (K) in our Mini-Chess cannot move, nor can it capture an attacking piece.

Regular chess rules:

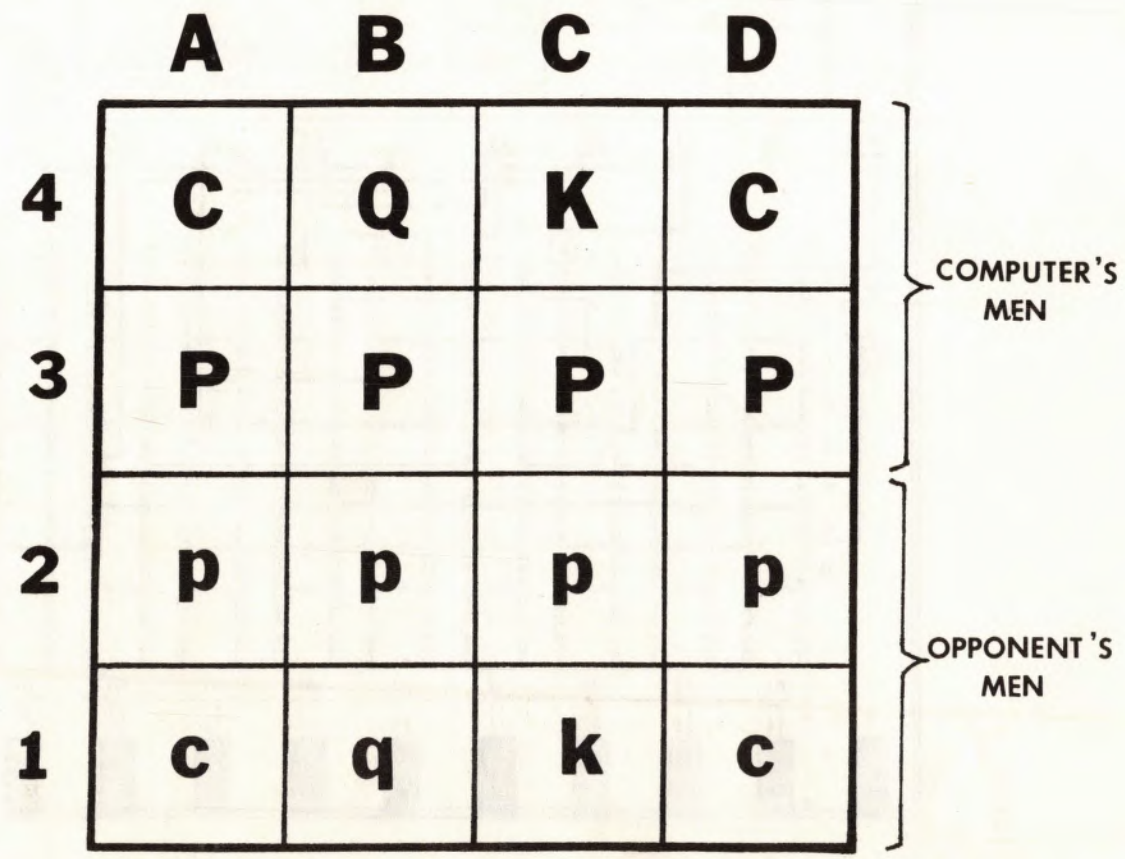
1. A chessman captures an opponent's piece by moving to the square on which the opposing piece is located and removing it.

2. The castle (C) moves vertically or horizontally.

3. The queen (Q) moves horizontally, vertically or diagonally.

4. The pawn (P) moves vertically, except when it captures an opposing piece, in which case it moves one square diagonally forward.

5. The object of the game is to threaten (check) the

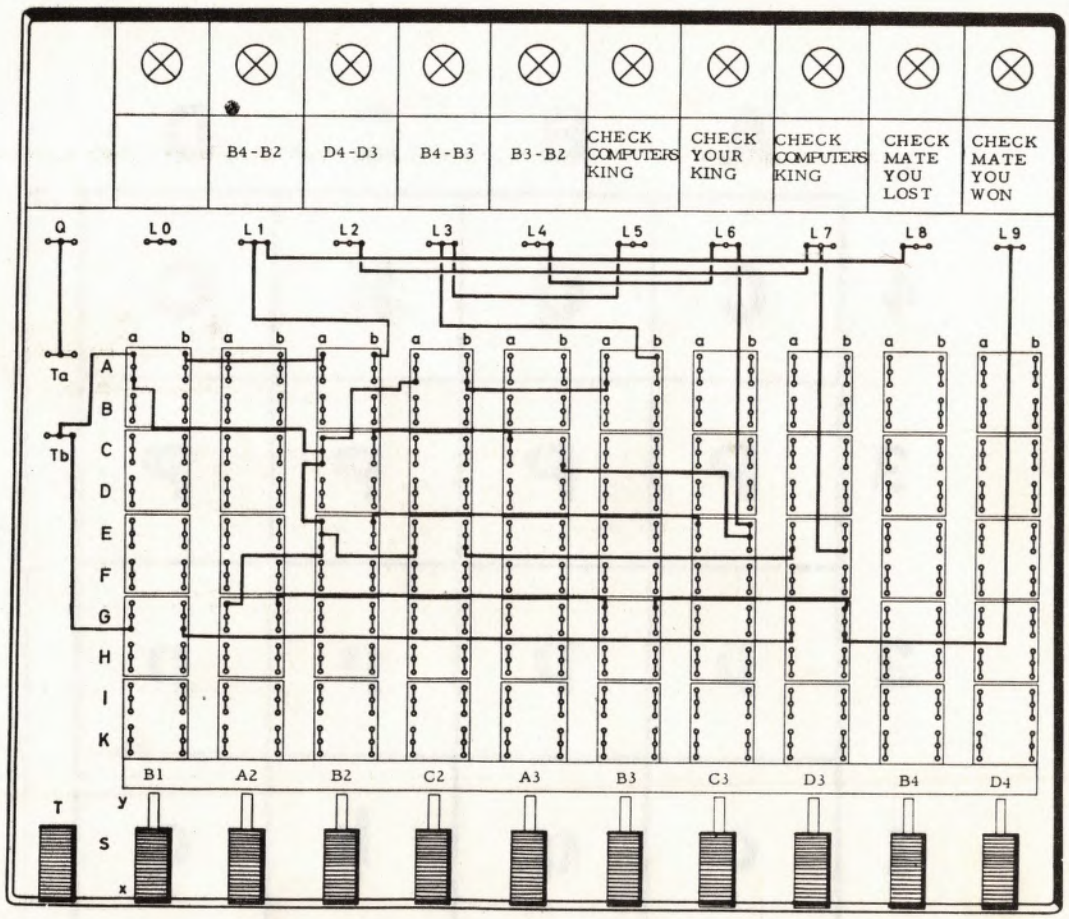


king in such a manner that the attacking piece cannot be captured by the opponent: that is - the king is doomed (i.e. checkmate).

In playing the game against the Miniature Computer, move the sliders corresponding to the origin and destination of the piece you wish to move. The sliders are normally in the x position; so when you move a piece, move the appropiate sliders to the y position. The computer's response will show on the lighted panel.

For example, if your first move is pawn at B2 taking the Computer's pawn at A3, push the sliders for B2 and A3 to the y position. Note that a light shows on the panel giving the computer's move as B3 to B2, that is - the pawn in front of the computer's queen advances one square, in doing so, the pawn "checks" your king and this too is indicated on the lighted panel.

The sliders should now be returned to the x position while you ponder your next move. Good luck!



WIRING SCHEME 41

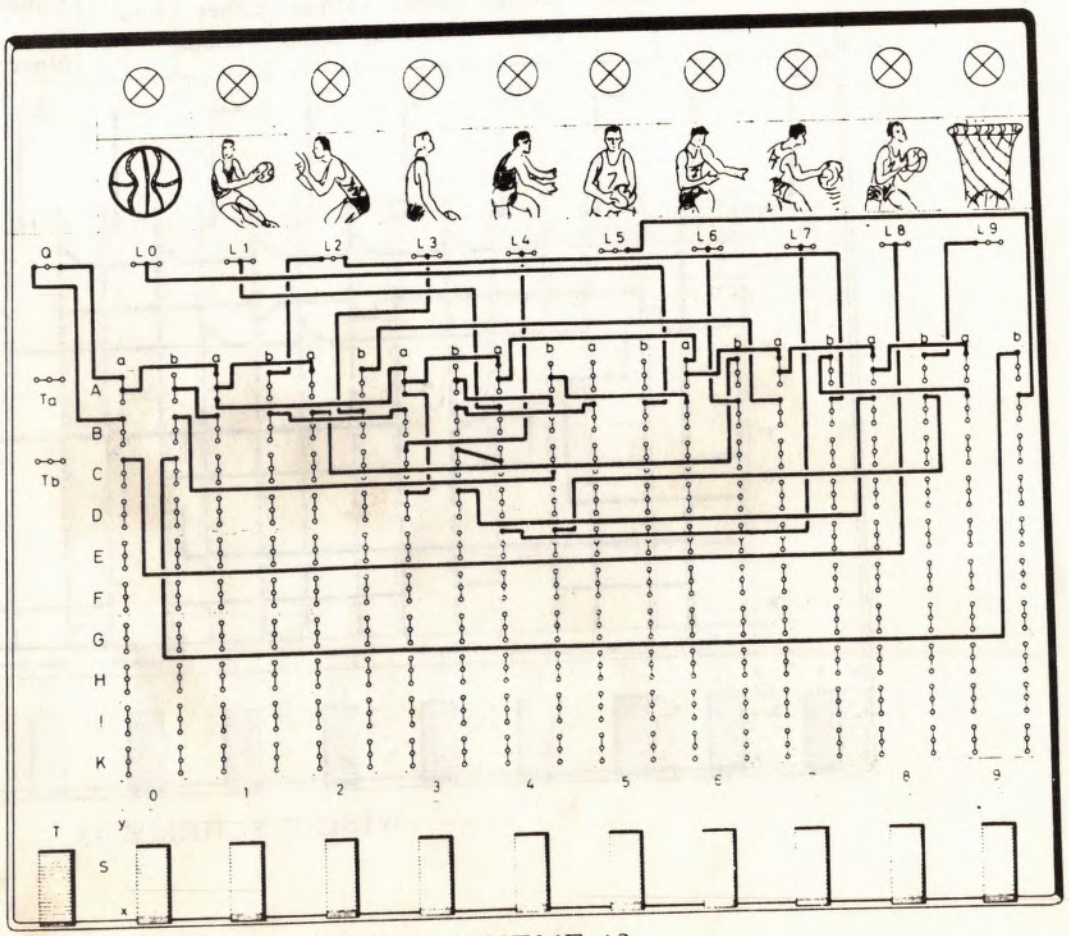
PROGRAM 42 - BASKETBALL

-Use transparent sheet No. 42.

Here is a way to participate in an exciting game of basketball without straining a single muscle! Complete the wiring according to Wiring Scheme 42 and insert the transparent sheet into the Output Section. Remember to mark the wiring with a colored pencil. This makes it easier to keep track.

To start the game, all sliders should be at X, except for Slider 0, which you move to y. Your "basketball" appears in the first compartment. The players can now take turns, each moving three sliders in either direction. To "put the ball through the basket" you have to get the last bulb to light up alone. The player that does this scores a goal and receives two points. After each goal, we begin again from the original position, with only Slider 0 at y. The game has ended when the first player reaches the 21st point.

Here is a special rule: if two bulbs light up at the same time, or, if all bulbs go out at once, it is a foul and a point is lost. The player must return the slider he has just moved to its original position, and give up his turn to the next player. You may find it difficult to score in this game of computer basketball, you may lose your patience, but at least you won't lose your breath!



WIRING SCHEME 42

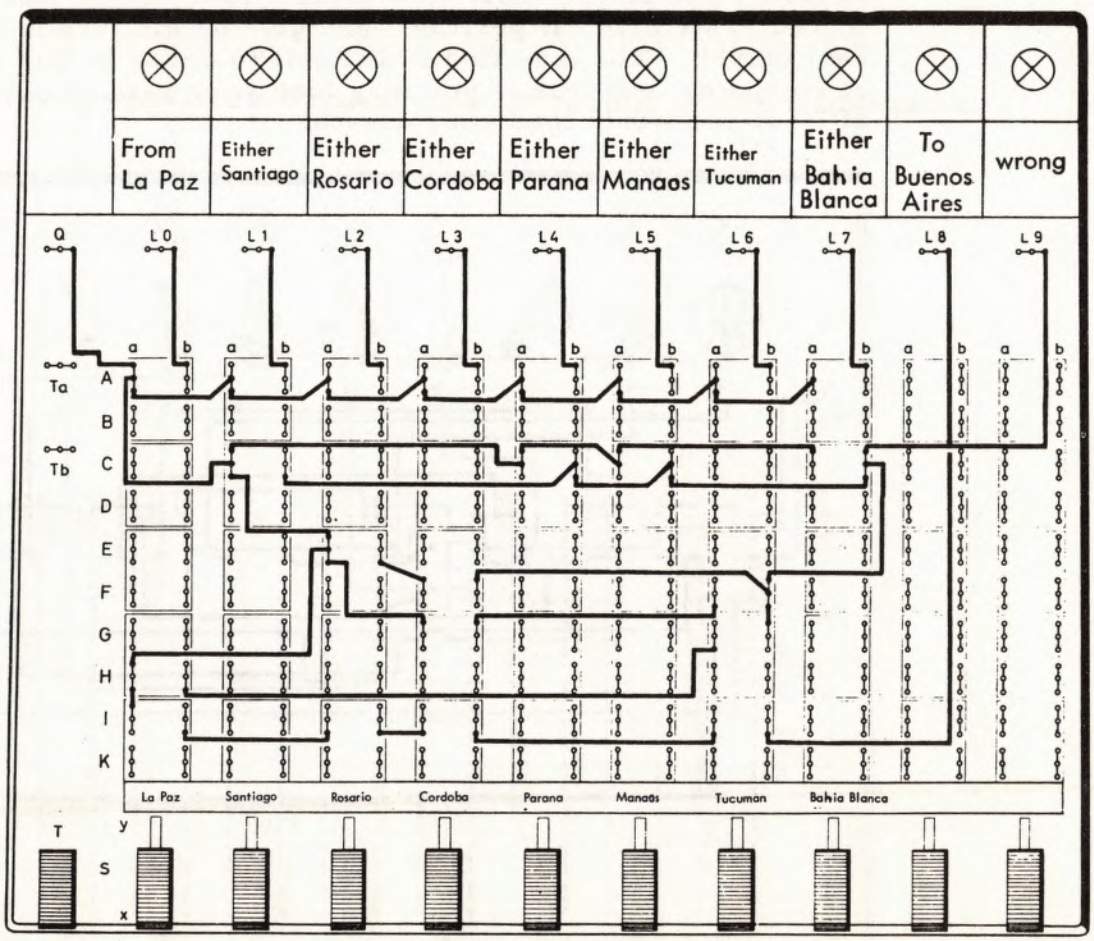
PROGRAM 43 - FROM LA PAZ TO BUENOS AIRES

-Use transparent sheet No. 43.

How do you find your way in a part of the world that you have never visited before? Your Miniature Computer can help you. Suppose you have to find your way from La Paz in Bolivia to Buenos Aires in Argentina. The question is: through which cities will you be able to reach your destination fastest?

First put Slider 0 to y. The indicator for La Paz will light up. Then, look for the first city on the way to Buenos Aires, by trying out another switch. If the indicator "Wrong" lights up together with that city, you have chosen the wrong route. When the correct city has been found, you leave its slider at y. The search goes on until all the cities have been found.

When the last correct city has been set, the indicator to Buenos Aires will light up. The problem has been solved.



WIRING SCHEME 43

PROGRAM 44 - MUSICAL CHORDS

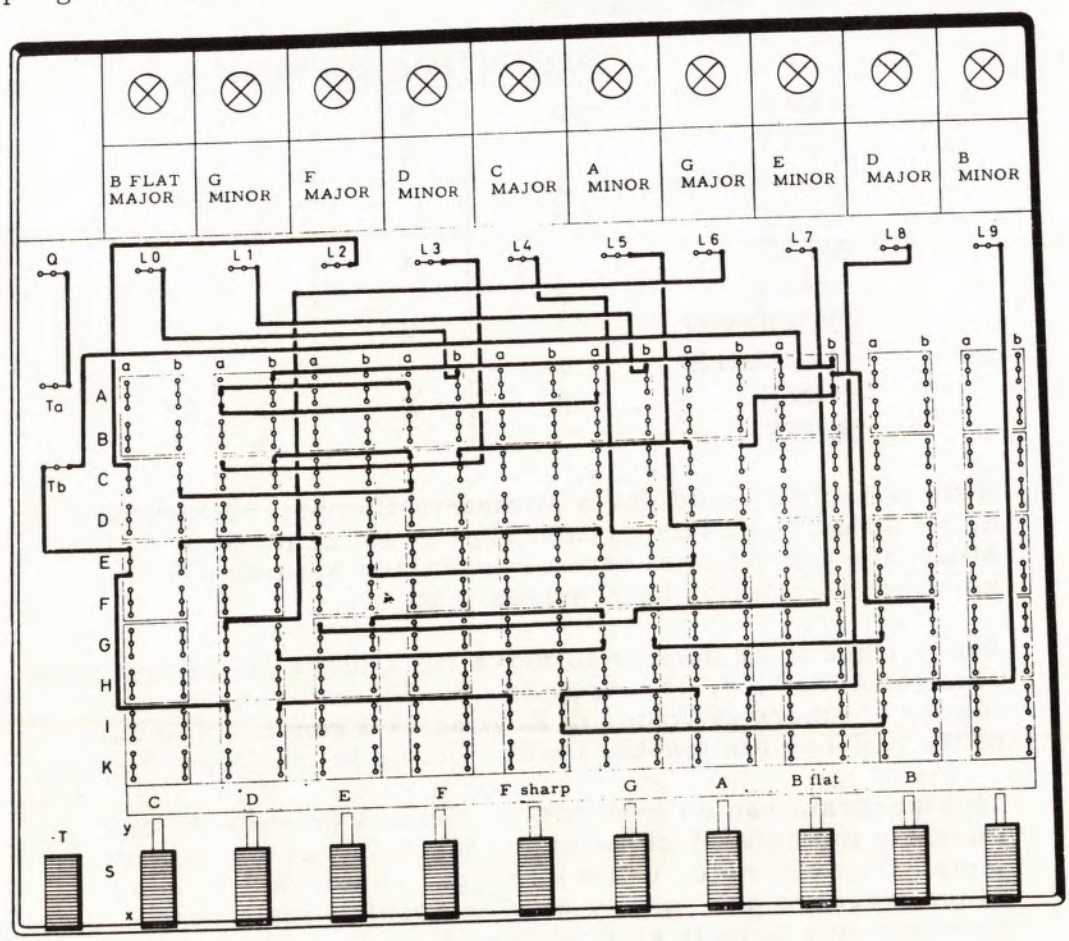
- Use transparent sheet No. 44.

Here is a program layout which will help us in learning musical harmony. We set the notes of a triad (chord of three notes) on the correct sliders and the lighted panel will indicate which triad (if any) has been selected.

Every major chord has a relative minor and these have been arranged in pairs on the light panel. Thus, we have the B flat major triad consisting of notes: B flat, D and F. Its relative minor triad, the G minor triad, is shown immediately next on the lighted panel. (The G minor triad consists of the notes: G, B flat and D.)

To help you in hearing these sounds, take your Miniature Computer to your piano, or other musical instrument. Push the correct sliders to light up the triad you want and then play the corresponding notes on the musical instrument to hear what it sounds like.

This is more fun than routine musical practice. Can you program other chords?

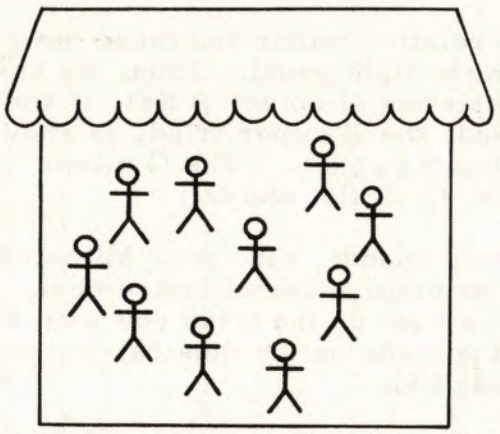


WIRING SCHEME 44

PROGRAM 45 - STUDYING THREE LANGUAGES

-Use transparent sheet No. 45.

Our new program deals with the "language department" of a large high school, where three languages (English, French and Spanish) are taught. There are ten teachers in the department. Most of them teach only one language, but others two or even three.



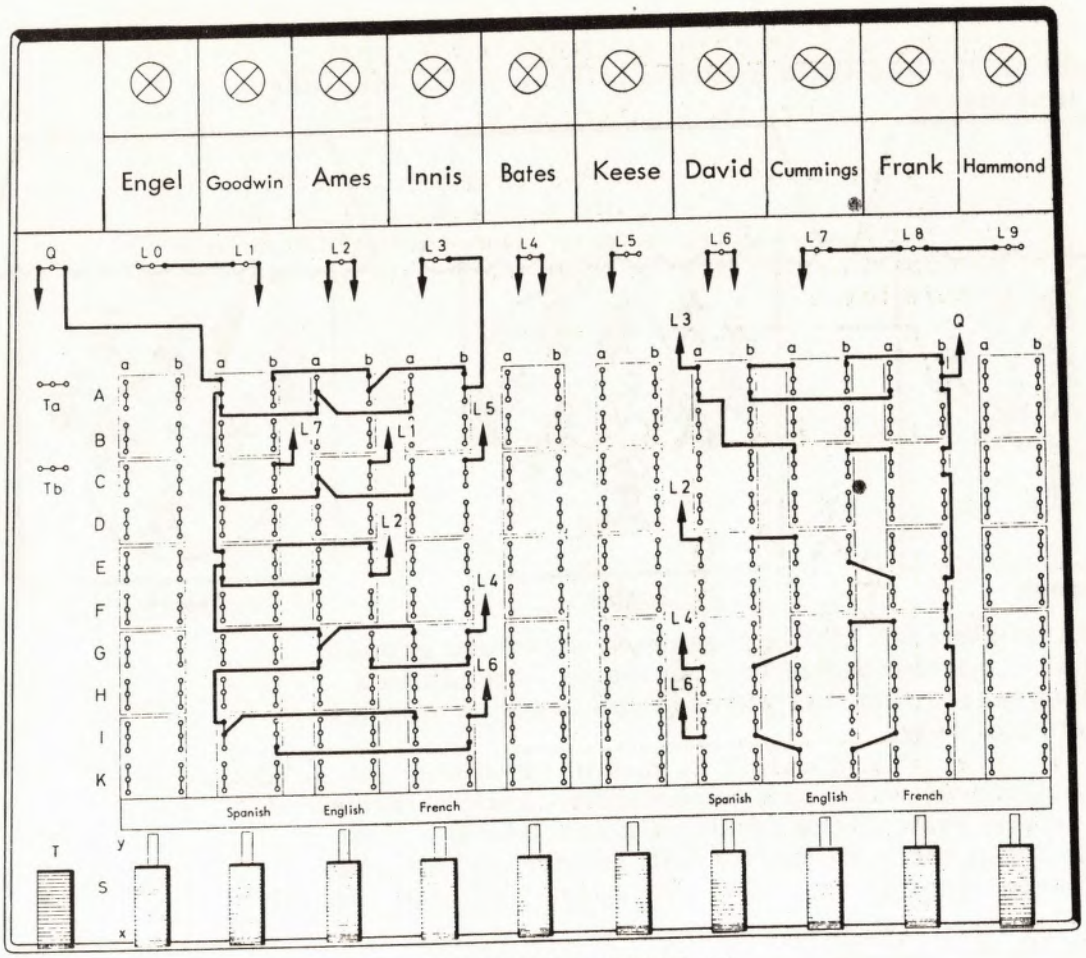
To make things easier in this department, the school principal has made up a chart which clearly shows which teachers teach what languages.

	SPANISH	ENGLISH	FRENCH
AMES	X	X	
BATES		X	X
CUMMINGS	X		
DAVID	X		X
ENGEL		X	
FRANK	X		
GOODWIN		X	
HAMMOND	X		
INNIS	X	X	X
KEESE			X

If the principal could use a Miniature Computer like yours, he would not need this written chart. He could wire up his Miniature Computer according to Wiring Scheme 45, to give the information he requires instantly.

All that has to be done is to move the sliders on the left hand side (Sliders 1, 2, 3) depending on whether you are looking for teachers of Spanish, English or French. The names of the appropriate teachers will light up in the Ouput Section.

This program can do even more. Wiring Scheme 45 can instantly indicate which teachers teach two or more subjects. For example, if the principal would want to know which teachers are equally good in Spanish and French, he would only have to shift Sliders 6 and 8. The appropriate bulb in the Output Section would show the required information.

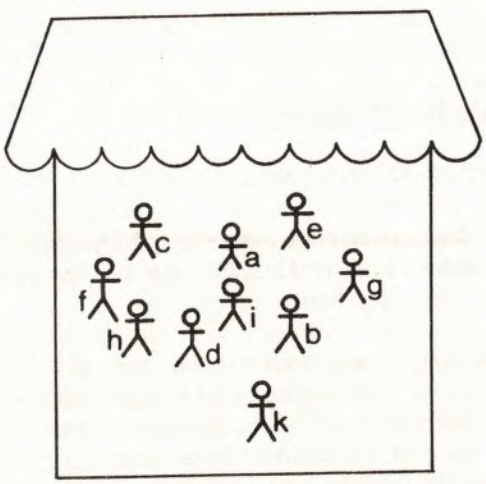


WIRING SCHEME 45

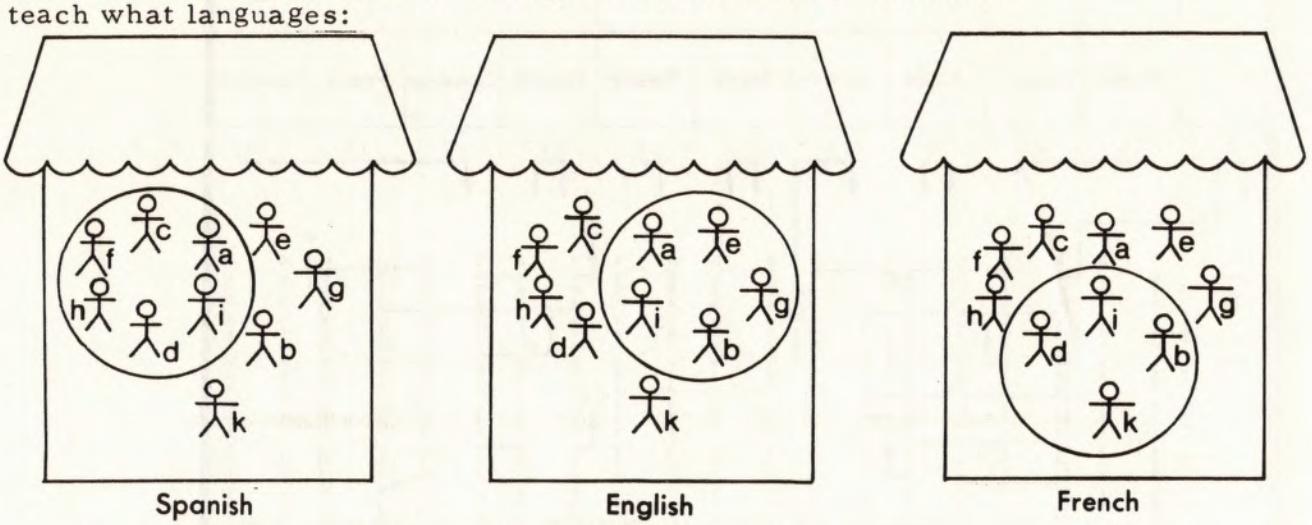
Background 1

CIRCLING THE TEACHERS

Now that we know a little about the teachers - we know all their names - it will be easy to improve our little diagram of the school by adding the first letter of their names to the little figures representing them.



Now we could circle this little diagram to show which teachers teach what languages:



The teachers within each circle represent a certain group or set, that is - the number of teachers able to teach in a particular language. And this observation leads us into the field of Algebra of Sets.

If we want to discover which teachers belong to which group, we can use an algebraic "language". We can take the first letters of each teacher's name in small letters and arrange them in brackets like this:

$$\{a, c, d, f, h, i\}$$

This represents the set (collection) of teachers teaching Spanish. Each teacher is shown as an "element" of the set concerned. We can now express the sets of teachers that teach each of our three languages (where S = Spanish, E = English and F = French) like this:

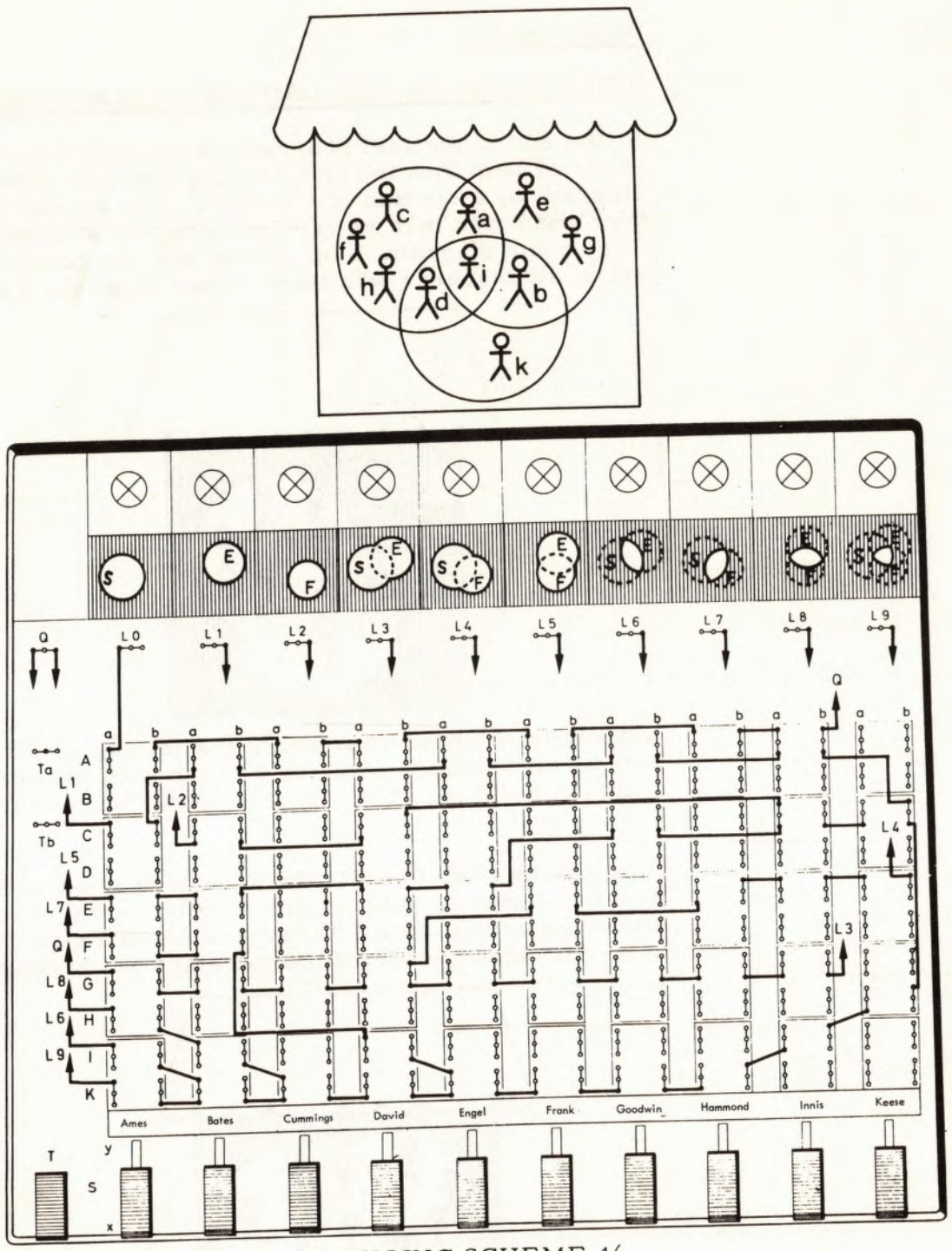
$$S = \{a, c, d, f, h, i\}$$
$$E = \{a, b, e, g, i\}$$
$$F = \{b, d, i, k\}$$

PROGRAM 46 - OVERLAPPING QUANTITIES

- Use transparent sheet No. 46.

We have circled the teachers of our three languages separately. Now we can join the three diagrams into a single one. Naturally the circles for the language groups sometimes overlap.

In this diagram, you can still clearly see which teachers teach what languages. We now have a program for our Miniature Computer which shows what area of our three circles each group of teachers belongs to. It is important that the sliders for all the teachers of each group are set to y - otherwise this program won't work.



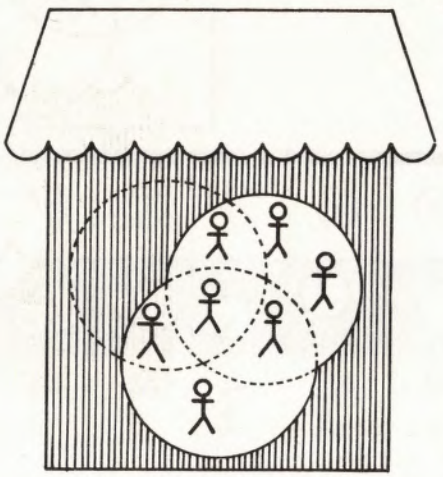
WIRING SCHEME 46

Here the circles overlap again. Sometimes, only sections of a circle will light up. If, for example, you move Sliders 3 and 8 forward (the ones for David and Innis), Light 7 will go on. This shows only the small segment where S and F overlap. If you think about it, this is only logical since David and Innis make up the small set of those teachers who teach Spanish and French.

Background 2

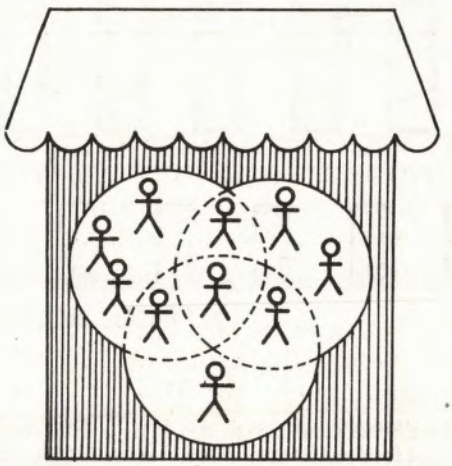
UNIFICATION OF THE LANGUAGE TEACHERS

At the end of the last program, we saw that two sets (the teachers of both Spanish and French) could be represented in our diagram by a section of a circle. Now we can go a little further and learn that we can join other sets in this way and show this in our diagram. Let us see how we could represent the set of all teachers who teach either English, French or both.



The diagram above represents the joining or "Union" of the sets of all teachers teaching French with the set of all those teaching English. The two sets were "united" to form a "Union set".

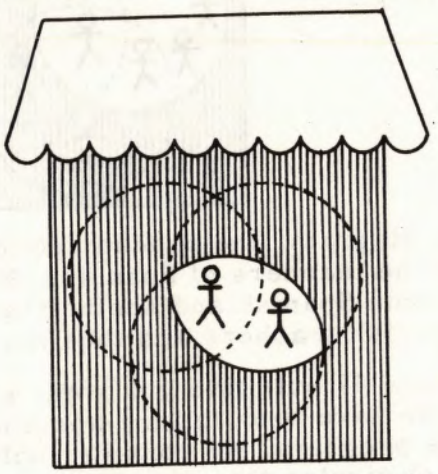
It is just as easy to unite the sets of Spanish and English teachers as those of Spanish and French teachers. Program 46 demonstrated this. There is also no reason why we cannot unite all three groups of teachers and get a diagram like this:



This Union of all the language teachers of the school represents a total group since it shows all of the teachers involved in this particular example. In other problems the Algebra of Sets, this total group could deal with the group of people in a room, in a country, the collection of thumbtacks on a bulletin board, the collection of molecules or any other group. In any case, this total group or set

represents all of the items in a particular problem and therefore this set is known as the Universal Set or more simply, the Universe.

We have already learned how to show in a diagram the Union of all those teachers who teach English and all those who teach French. However, what if we are looking for a method to show those especially talented teachers who could teach both English and French? We do it like this:



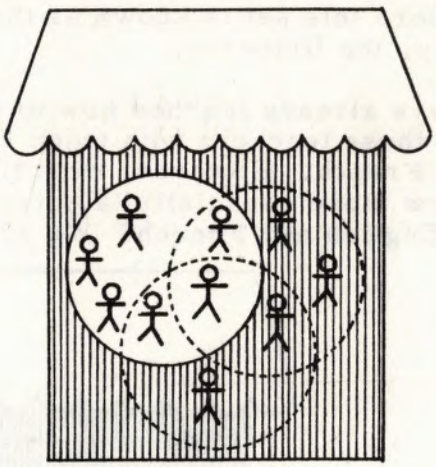
As you can see, the unshaded section of a circle above is made up both of a part of the circle for English teachers and part of the one for French teachers, that is - the parts which overlap. As a result, each teacher in the overlap is a member of both the French and English teaching groups. In our system of mathematics, this overlapping area is called the Intersection. Thus, the unshaded area represents the Intersection of circles E and F.

PROGRAM 47 - THE COMPLEMENT

-Use transparent sheet No. 47.

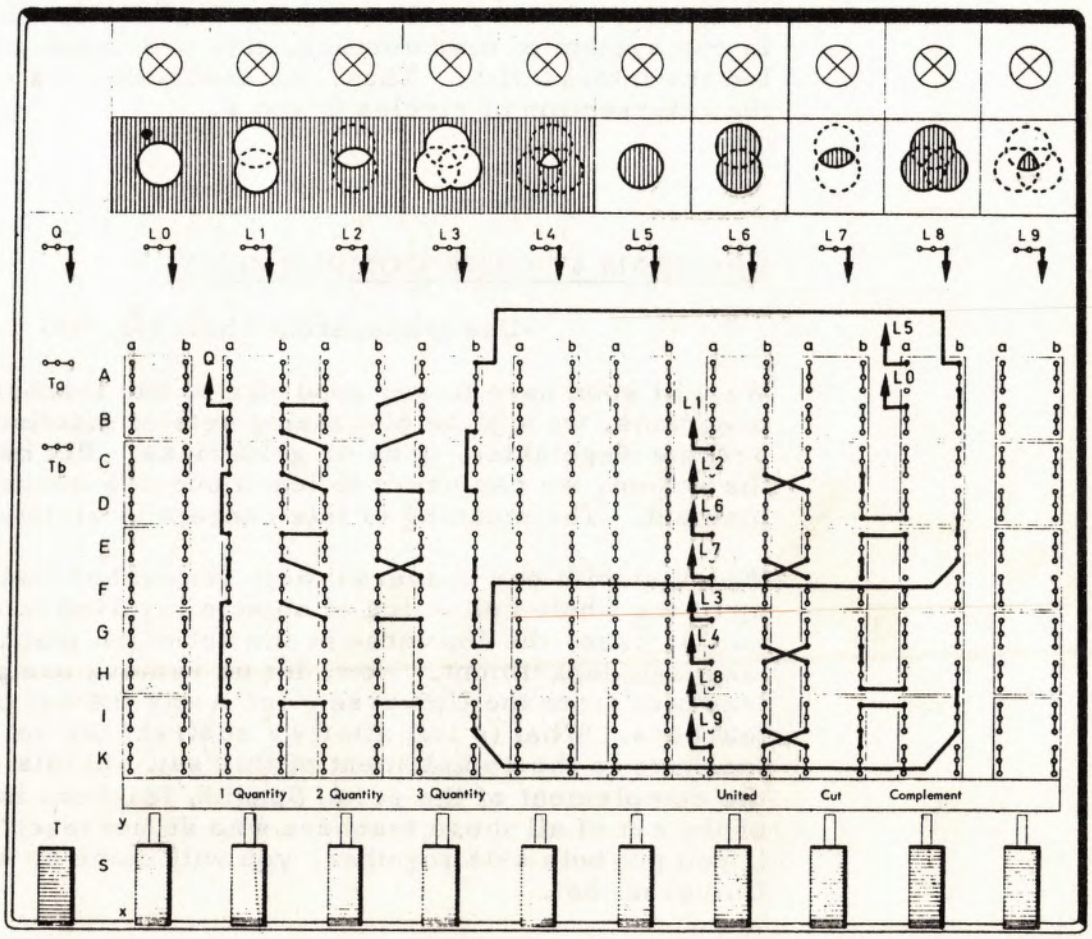
We will soon have to say good-bye to our teachers. In future programs, we will be discussing sets of indefinite members - perhaps vegetables, guns or goldbricks. But before we leave the school, we can use it to learn one new concept: the Complement. The meaning of this concept is simple.

We start with our Universal Set; remember that this is made up of the whole collection of objects involved in a problem. In this case, the Universe is the set of all teachers in the language department. Now, let us remove one group of teachers from the Universe - let's say the set of Spanish teachers. What is left after we subtract the set of Spanish teachers is the complement of this set. In this example, the complement of the set of Spanish teachers is made up of the set of all those teachers who do not teach Spanish since, if you put both sets together, you will come up with the whole, Universal Set.



In our diagram, the shaded section represents the complement of the teachers of Spanish. Note that the teachers who teach both Spanish and another language are part of the circle for Spanish teachers and thus not part of the complement.

Program 47 will give you some practice in understanding what we have been talking about up to this point. Note that in this program, we consider only the circles or the sections of circles that are lit up, not the shaded areas. After this program, to make it more simple, we are going to reverse things. We will consider only the shaded areas as being part of our results.



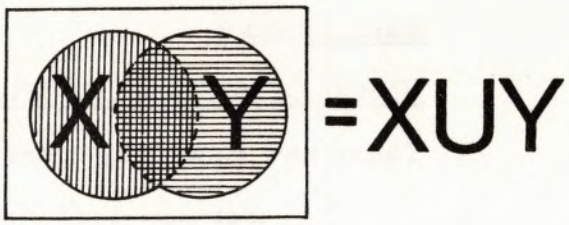
WIRING SCHEME 47

Background 3

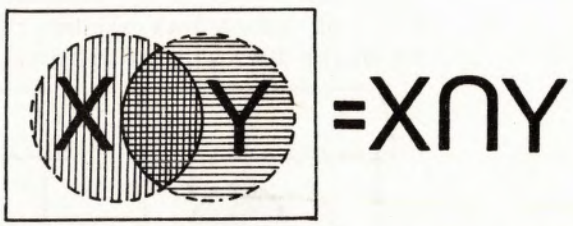
THE FIRST LAWS

Now we are ready to leave the high school. Instead of considering sets of French, English and Spanish teachers, we will deal with sets simply called X, Y and Z. We can still do some very interesting things with them. Also, you might note that you do not have to consider a universe of three sets. We may consider one set, thirty or a hundred sets at a time.

We are now going to introduce you to two new symbols so that you can work more easily with sets. A little while back, we talked about the Union of two sets. In the next diagram, we have two united sets: X (shaded vertically) and Y (shaded horizontally). Rather than taking the trouble of writing out "set X is united with set Y" over and over again, we simply write: XUY.



We also have another symbol ($\cap$) to represent the Intersection of two sets. Instead of saying "set X intersects with set Y", we write simply $X \cap Y$.

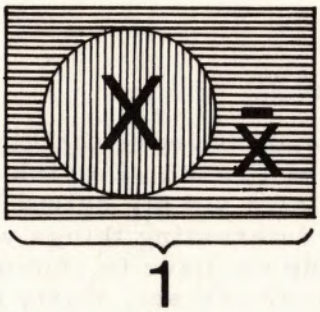


There is a third symbol that we must learn. To represent the Complement of a set, such as the Complement of x, we simply put a horizontal line over the original set, like this $\overline{X}$. Thus $\overline{A}$ would be the Complement of A and $\overline{C}$ would be the Complement of C.

We are now ready to meet our first Law of sets - ideas so simple that most people would call them obvious. This first Law, stated in words, is as follows: the Union of a set and its Complement is the Universal Set. Using our new symbols, we can express this Law as follows:

$$X \cup \overline{X} = 1$$

You may be a bit stumped by the 1 in this question. The 1 simply represents the Universal Set. So now you have a fourth symbol to remember:



$X \cup \overline{X} = 1$

And now we come to two more Set Laws.

Set Law 1: $X \cup Y = Y \cup X$
Set Law 2: $X \cap Y = Y \cap X$

These Set Laws can be compared to the Law of Arithmetic that states that $2 + 3 = 3 + 2$.

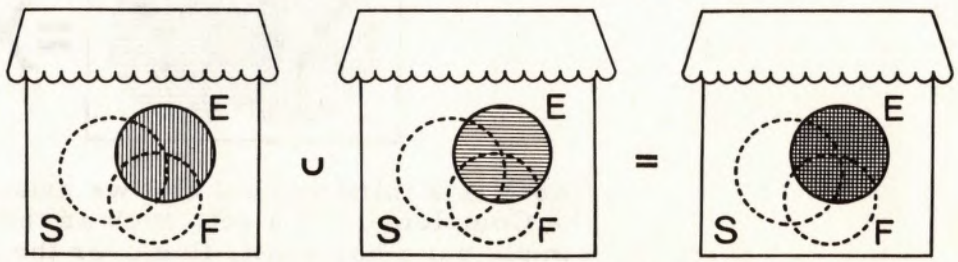
Background 4

UNION OF SETS WITH THEMSELVES

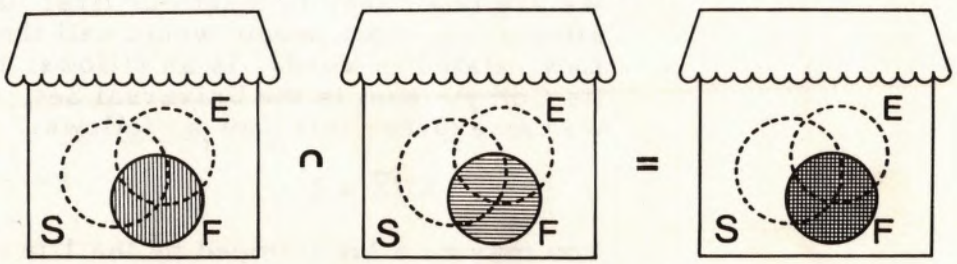
These new Laws might seem a little strange at first:

Set Law 3: $X \cup X = X$
Set Law 4: $X \cap X = X$

If you think about them a little, they will start to seem more sensible. For instance, if we unite the set of English teachers with the set of English teachers, we are adding two sets, each of which has all the members of the other and neither of which has any additional members. Thus $E \cup E = E$.



It is the same in the case of the other Law. The Intersection of the two identical sets consists of all the members common to both, that is all the members of the set. Thus $E \cap E = E$.

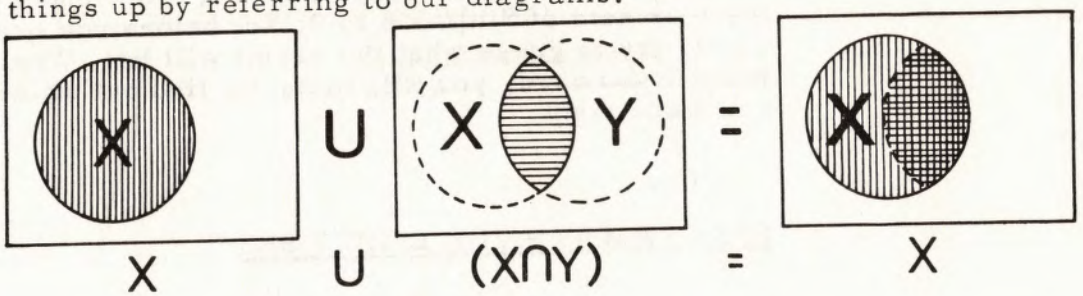


Now to make things a bit more complicated, here are two more Laws:

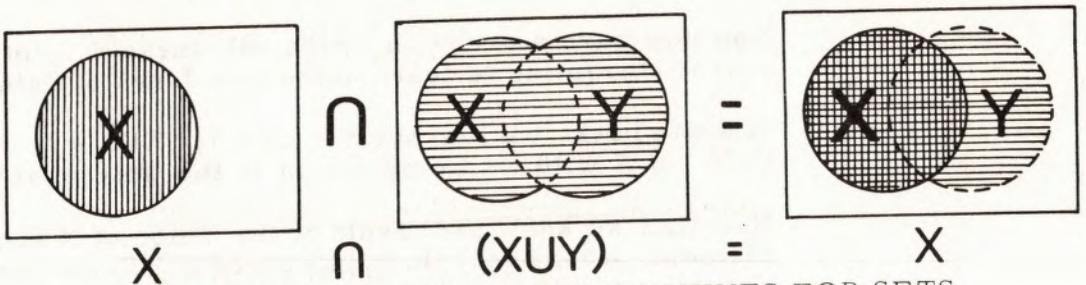
Set Law 5: $X \cup (X \cap Y) = X$
Set Law 6: $X \cap (X \cup Y) = X$

Note that for these Laws, whatever is written between brackets must first be worked out separately.

If these two Laws seem a bit strange, we can simply clear things up by referring to our diagrams:



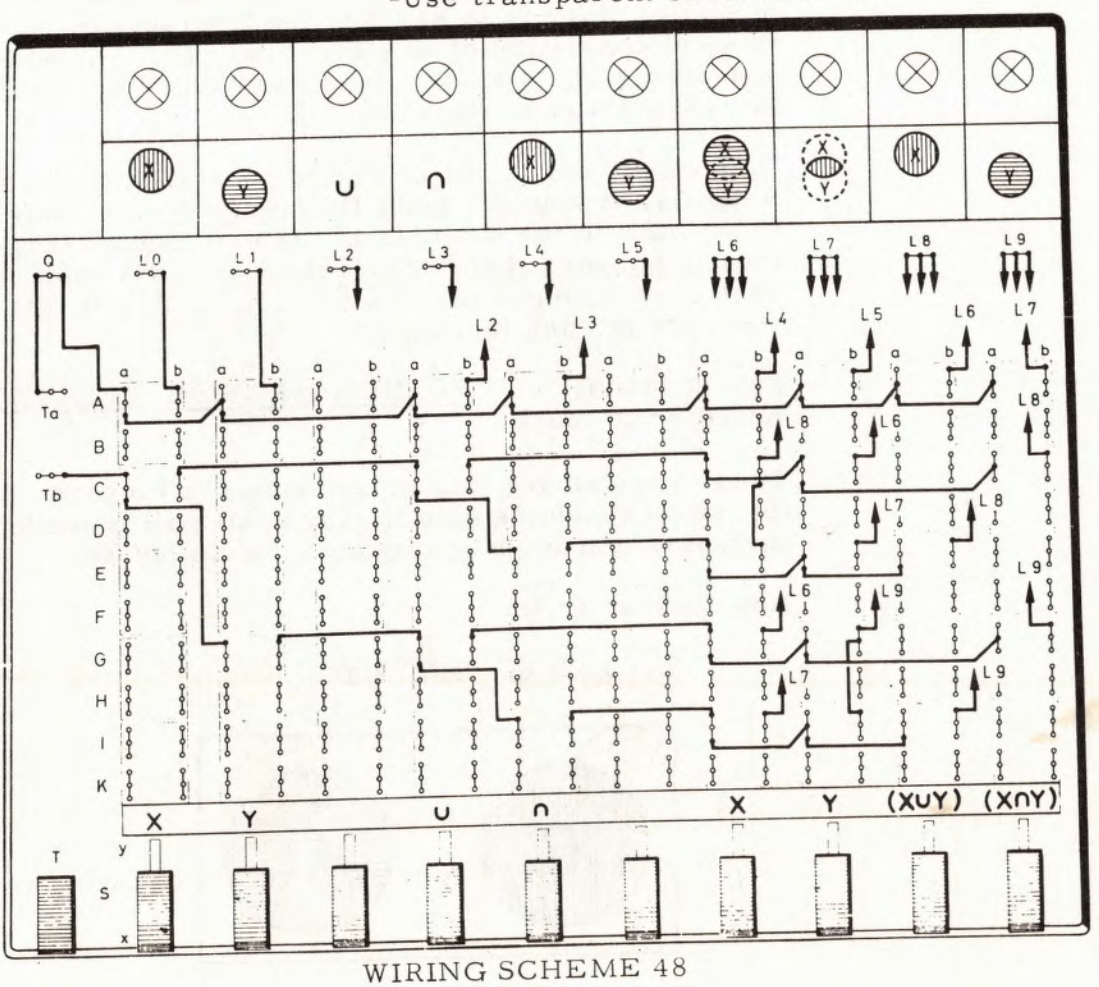
X ∪ (X∩Y) = X



X ∩ (XUY) = X

PROGRAM 48 - TEACHING MACHINES FOR SETS

-Use transparent sheet No. 48.



WIRING SCHEME 48

This "teaching machine" program will be great for getting yourself more used to some of the basic Laws of Sets. To use this program, simply start by pushing forward either X or Y on Sliders 0 and 1. Also set one of the symbols ($\cup$ or $\cap$) forward on Sliders 3 and 4. Now choose one of the four sets of Sliders 6 to 9; but before you press it forward, try to guess what the result will be! When you do move it forward, you will instantly find out whether or not you were right.

PROGRAM 49 - THE EMPTY SET

- Use transparent sheet No. 49.

You now have a program which will make it a lot easier and more interesting to learn some new Laws of Sets. .

A while back, we learned Set Law 7: $X \cup \overline{X} = 1$; that is, the Union of X with its Complement is the Universal Set.

Now that we know the result of the Union of X and its Complement, what about their Intersection? Since the Intersection of two sets is made up of the members common to both sets, it seems obvious that there are no members contained by both X and its Complement ($\overline{X}$). Another way of stating this, is to say that the Intersection of X and $\overline{X}$ is a set consisting of no members. This memberless set has been given a name. It is known as "the empty set" and is represented by the symbol $\emptyset$.

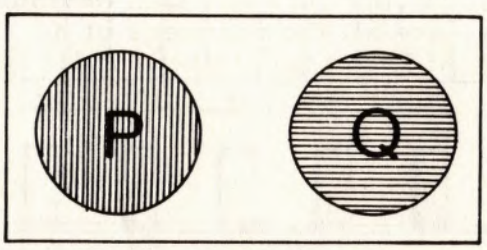
This empty set is not as mysterious as it might seem. Actually, it appears quite frequently in everyday life. Let us go back to the example of the high school language department. Suppose that the principal sent the head of the language department a form to fill out, listing the number of teachers of each language.

No. of teachers qualified to teach	FRENCH	SPANISH	ENGLISH	GERMAN
	4	6	5	0

Thus, you can see that in our school's language department, the set of teachers able to teach German is made up of no members and is an example of the Empty Set.

Now we can write:

Set Law 8: $X \cap \overline{X} = \emptyset$



Now let us look back at this Law concerning the Intersection of X and the Empty Set.

Set Law 9: $X \cap \emptyset = \emptyset$

This can be understood very easily if we remember that since the Empty Set has no members, X and the Empty Set cannot have any members in common.

This new Set will now seem quite logical:

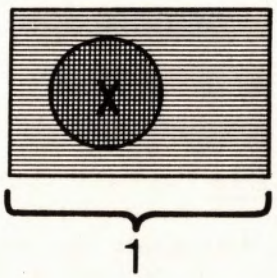
Set Law 10: $X \cup \emptyset = X$

Background 5

THE UNIVERSAL SET

Remember that the Universal Set (1) is the set that contains all the members of all the sets under consideration. For instance, in our example of the language department it is the set containing all the teachers of the department. In terms of our diagrams, the Universal Set is the entire area of our rectangle.

What happens when we unite X with the Universal Set? This is a situation a bit like the ones in which we added X to itself.



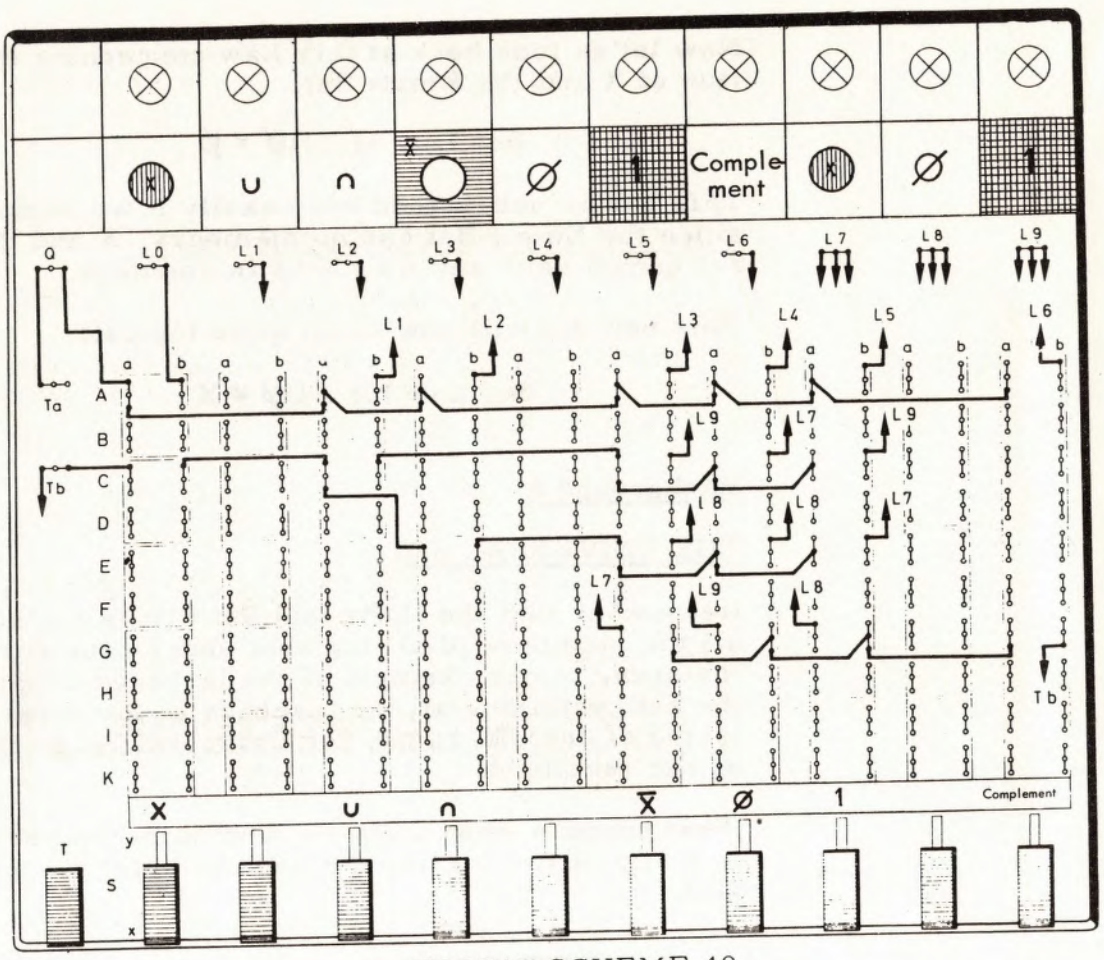
This time, we are uniting a part of the Universal Set (X) with the Universal Set. It is obvious that the result of this Union is the Universal Set. Thus:

Set Law 11: $X \cup 1 = 1$

Now what about the Intersection of X and the Universal Set? The Intersection is made up of all the members common to both sets and it is obvious that these common members are all the members of X. Therefore, the Intersection of X and the Universal Set is X.

Set Law 12: $X \cap 1 = X$

You can have some more practice with these new Laws in Program 49.



WIRING SCHEME 49

Background 6

COMPLEMENTARY SETS

There are three Laws that apply to Complementary Sets. All of these can be learned with the aid of Program 49. To find the Complementary Set of any of the sets on this program, set the sliders forward for that set as well as the one on the right marked "Complement". The Complementary Set will then automatically appear in the Output Section.

We have already seen that the Complementary Set of X is $\overline{X}$. We can also see that $\overline{X}$ has a Complementary Set and it is X. Thus, we can call X the Complementary Set of the Complementary Set ($\overline{\overline{X}}$).

Set Law 13: $\overline{(\overline{X})} = X$

To say Set Law 13 in words: the Complementary Set of the Complementary Set of X is X. Here are two more laws dealing with Complementary Sets that you should be able to understand easily.

Set Law 14: $\overline{I} = \emptyset$

This is quite clear because we defined the Complement of a set as the difference between that set and the Universal Set. Obviously, the difference between the Universal Set and the Universal Set is nothing, that is - the Empty Set.

The opposite is also true. What is missing from the Empty Set to form the Universal Set, is the Universal Set. Thus:

Set Law 15: $\overline{\emptyset} = 1$

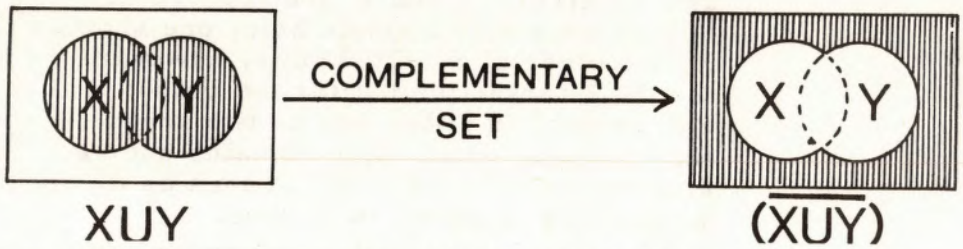
PROGRAM 50 - DE MORGAN'S LAWS

- Use transparent sheet No. 50.

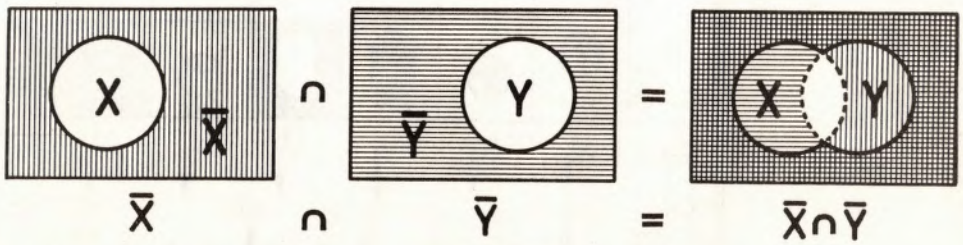
There are only two Laws of Sets that are a little hard to understand at first. They are names after their inventor, Augustus de Morgan.

Set Law 16: $(\overline{X \cup Y}) = \overline{X} \cap \overline{Y}$
Set Law 17: $(\overline{X \cap Y}) = \overline{X} \cup \overline{Y}$

We will understand these two more difficult Laws if we use our diagrams. First let us try $(\overline{X \cup Y}) = \overline{X} \cap \overline{Y}$.



This diagram lets you understand the left side of our equation. Now look at this:

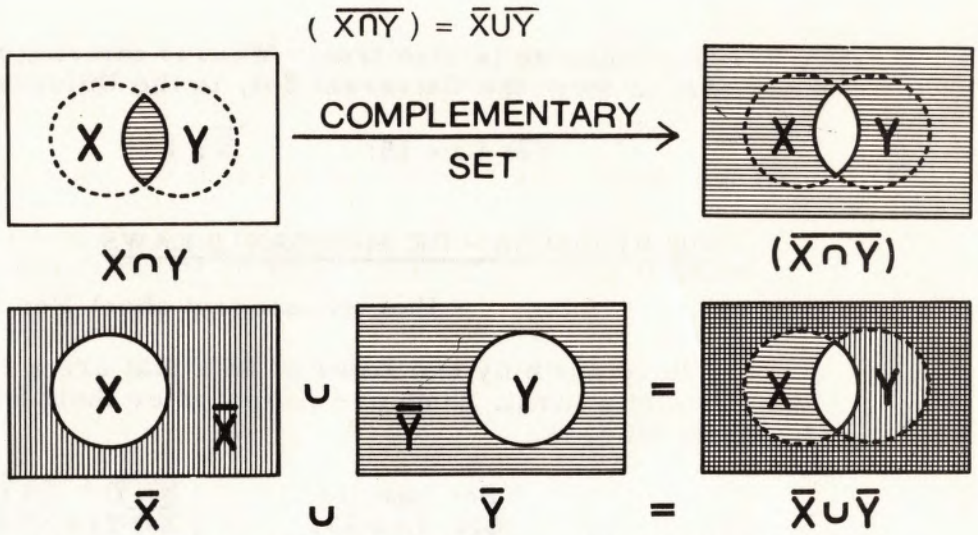


Now compare the diagrams for $(\overline{X \cup Y})$ and $\overline{X} \cap \overline{Y}$. It is clear that the areas of the shaded space of the first diagram and the doubly shaded area of the second diagram are equal.

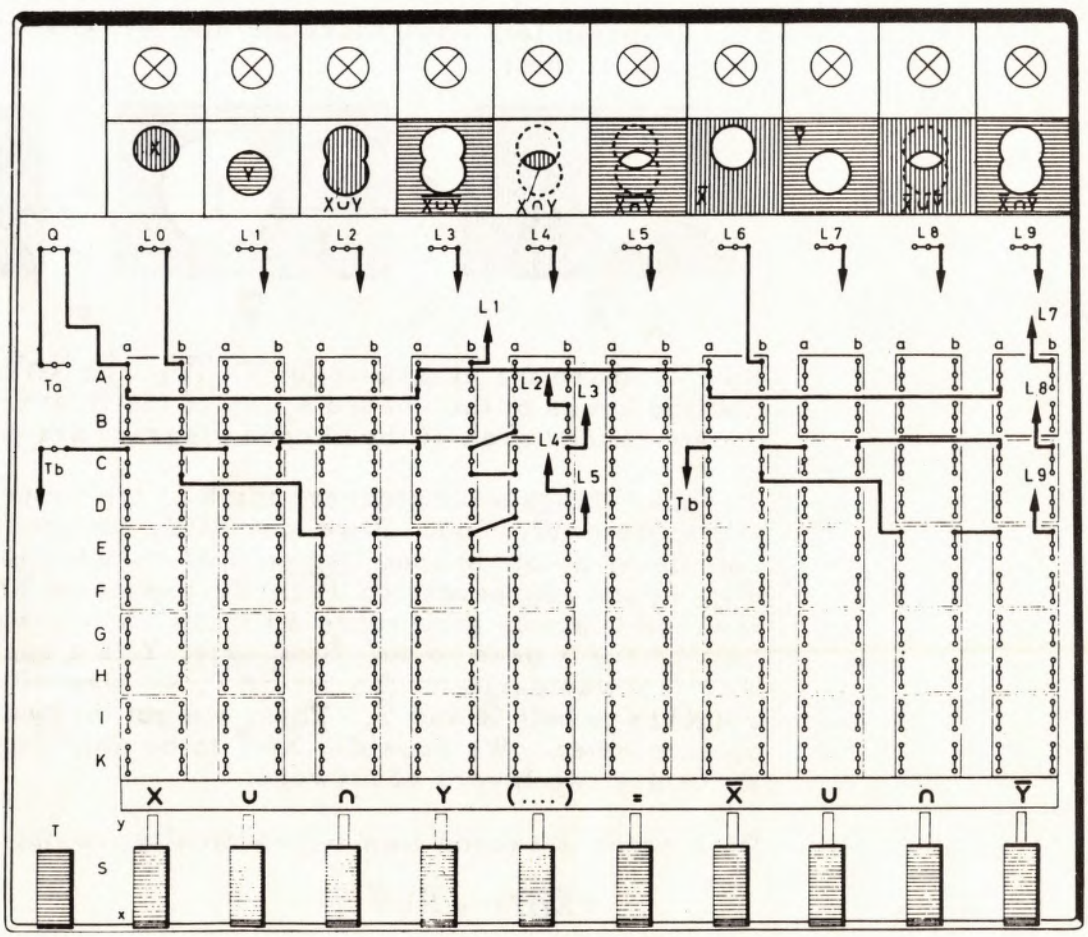
To make things a bit clearer, think of the Universal Set as a rectangle of plywood. If we saw XUY, we are left with a board containing a hole shaped like an oval indented in the center. This board represents $(\overline{X \cup Y})$. To represent $\overline{X} \cap \overline{Y}$ without our plywood board is a bit more difficult. X is a board with a hole in it where X used to be. Similarly, Y is a board with a hole where Y was earlier. To get $\overline{X} \cap \overline{Y}$, we want to get the areas common to both $\overline{X}$ and $\overline{Y}$. Thus, we put the two boards on top of each other. We consider $\overline{X} \cap \overline{Y}$ to be only these areas where there are two layers of plywood.

De Morgan's second Law is practically the opposite of the first:

$(\overline{X \cap Y}) = \overline{X} \cup \overline{Y}$



To understand this one, we should use our plywood boards again. (X∩Y) is a board where we have sawed out only the one little slice that represents the Intersection between X and Y (X∩Y). $\overline{X}$ and $\overline{Y}$ are represented just as before: two boards each with a single hole, one showing the removal of X, the other of Y. This time, however, we are not looking for the Intersection but for the Union; that is, it is the wood that counts, in either one or two areas. Here we finally see that only one small slice remains cut out. This makes this diagram exactly the same in area as the one for ($\overline{X \cap Y}$), and so our Law is shown to be true.



WIRING SCHEME 50

Now you can become more familiar with all these Laws by working with Program 50. With this program, you can reconstruct De Morgan's Laws in the form of equations.

You can set the left side of the equation on Sliders 0 to 4 and the right side on Sliders 6 to 9. To start, you can set the Union or Intersection of X and Y between Sliders 0 and 3. To make it clearer, X and Y light up in the Output Section when you do this. Now when you push the push button (T) , you will see a representation of the Union and Intersection in the Output Section.

If we want to represent De Morgan's Laws in this scheme, we must be able to show the Complements of these sets. To do this, we use Slider 4. When this slider is set and the push button is applied, the Complementary Sets will appear in the Output Section.

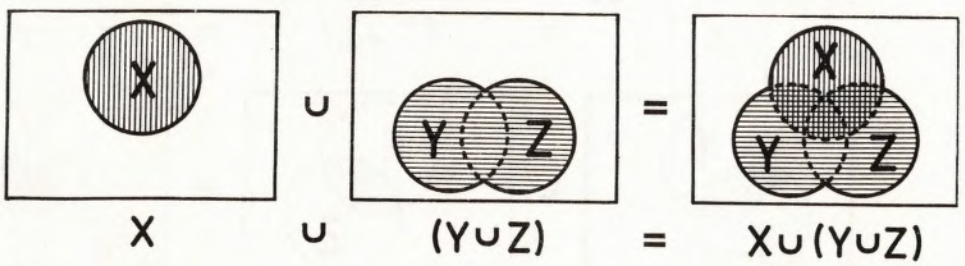
Now you can show the Union and the Intersection of $\overline{X}$ and $\overline{Y}$ on Sliders 6 to 9. If you press the push button and the same figures light up for both the left and right side, the equation will be correct.

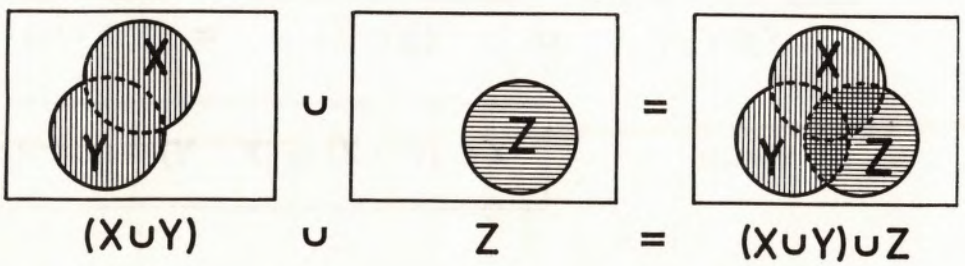
Background 7

THE LAST FOUR LAWS

There are a total of twenty-one Set Laws. We have only four more to consider before completely understanding the subject.

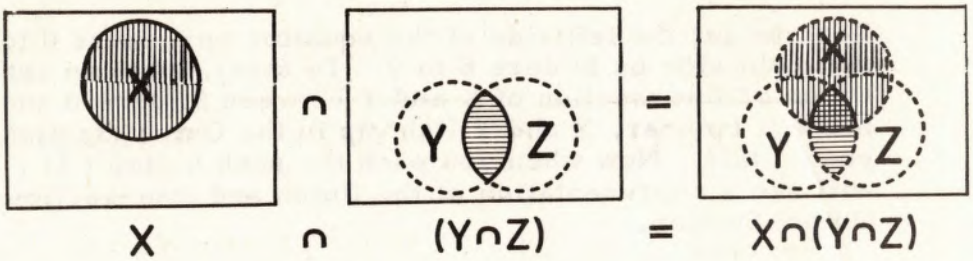
These last four are quite simple. You will pick them up quickly just by studying the following diagrams.



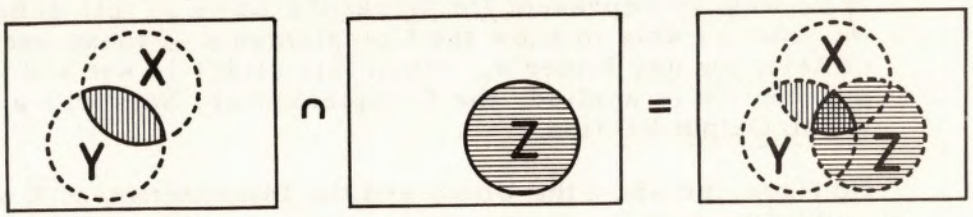


Set law 18

$$X \cup (Y \cup Z) = (X \cup Y) \cup Z$$



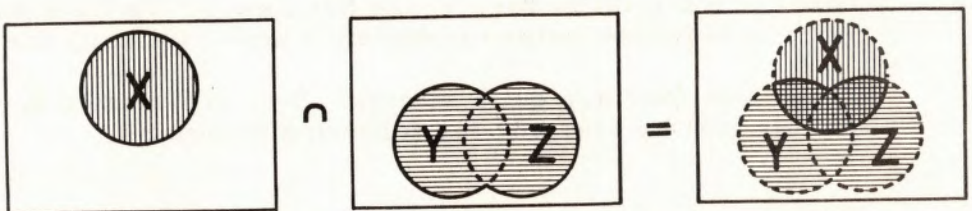
$X \cap (Y \cap Z) = X \cap (Y \cap Z)$



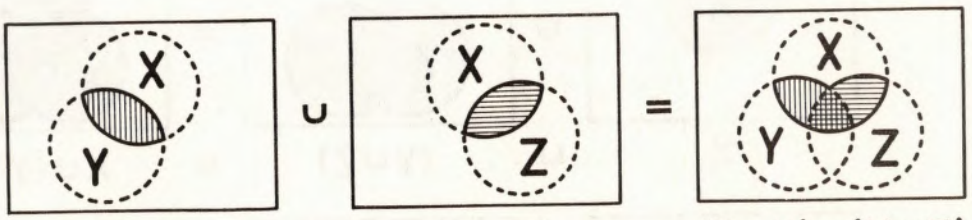
$(X \cap Y) \cap Z = (X \cap Y) \cap Z$

Set law 19

$$X \cap (Y \cap Z) = (X \cap Y) \cap Z$$



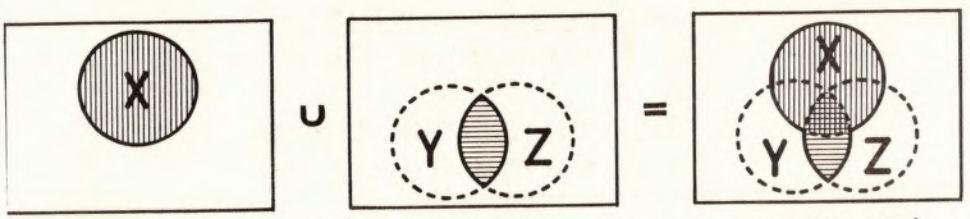
$X \cap (Y \cup Z) = X \cap (Y \cup Z)$



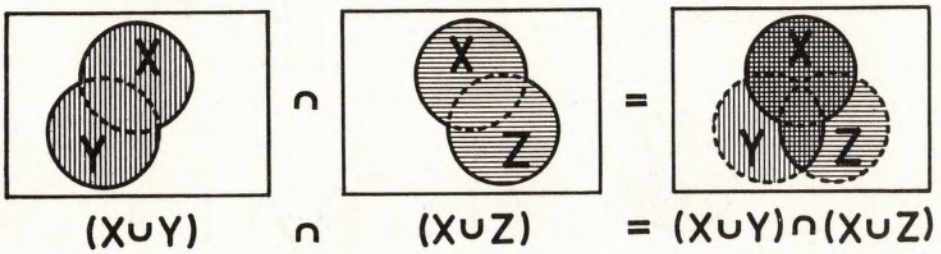
$(X \cap Y) \cup (X \cap Z) = (X \cap Y) \cup (X \cap Z)$

Set law 20

$$X \cap (Y \cup Z) = (X \cap Y) \cup (X \cap Z).$$



$X \cup (Y \cap Z) = X \cup (Y \cap Z)$



Set law 21 $X \cup (Y \cap Z) = (X \cup Y) \cap (X \cup Z)$

Background 8

SUMMARY OF SET LAWS

Here, for the sake of order and completeness, is a list of all the Laws of Sets.

Now you are ready to make some practical use of these Laws in programming your Miniature Computer!

Set law (1) $X \cup Y = Y \cup X$
(2) $X \cap Y = Y \cap X$

(3) $X \cup X = X$
(4) $X \cap X = X$

(5) $X \cup (X \cap Y) = X$
(6) $X \cap (X \cup Y) = X$

(7) $X \cup \overline{X} = 1$
(8) $X \cap \overline{X} = \emptyset$

(9) $X \cap \emptyset = \emptyset$
(10) $X \cup \emptyset = X$
(11) $X \cup 1 = 1$
(12) $X \cap 1 = X$

(13) $(\overline{\overline{X}}) = X$

(14) $\overline{1} = \emptyset$
(15) $\overline{\emptyset} = 1$

(16) $\overline{X \cup Y} = \overline{X} \cap \overline{Y}$
(17) $\overline{X \cap Y} = \overline{X} \cup \overline{Y}$

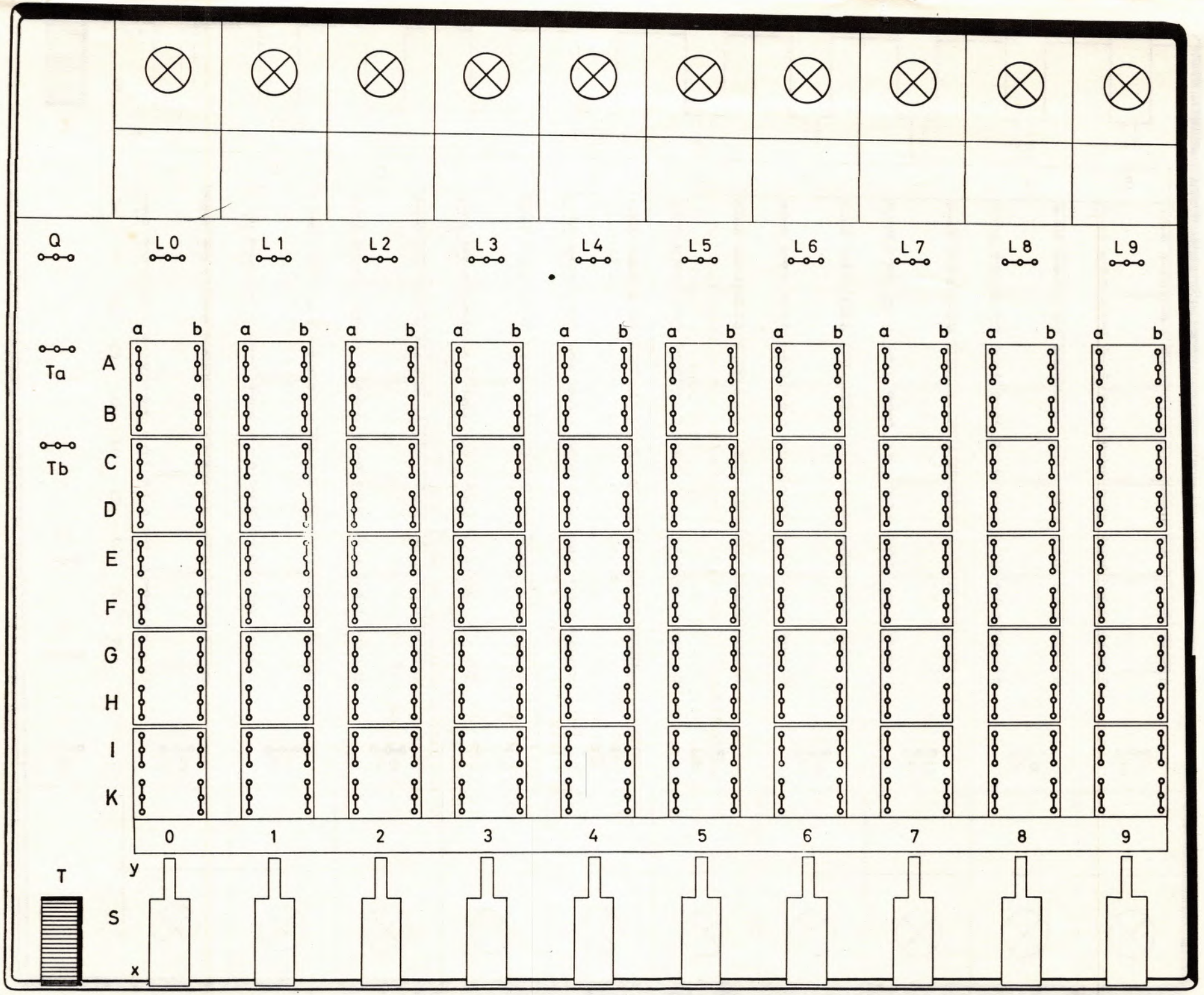
(18) $X \cup (Y \cup Z) = (X \cup Y) \cup Z$
(19) $X \cap (Y \cap Z) = (X \cap Y) \cap Z$

(20) $X \cap (Y \cup Z) = (X \cap Y) \cup (X \cap Z)$
(21) $X \cup (Y \cap Z) = (X \cup Y) \cap (X \cup Z)$

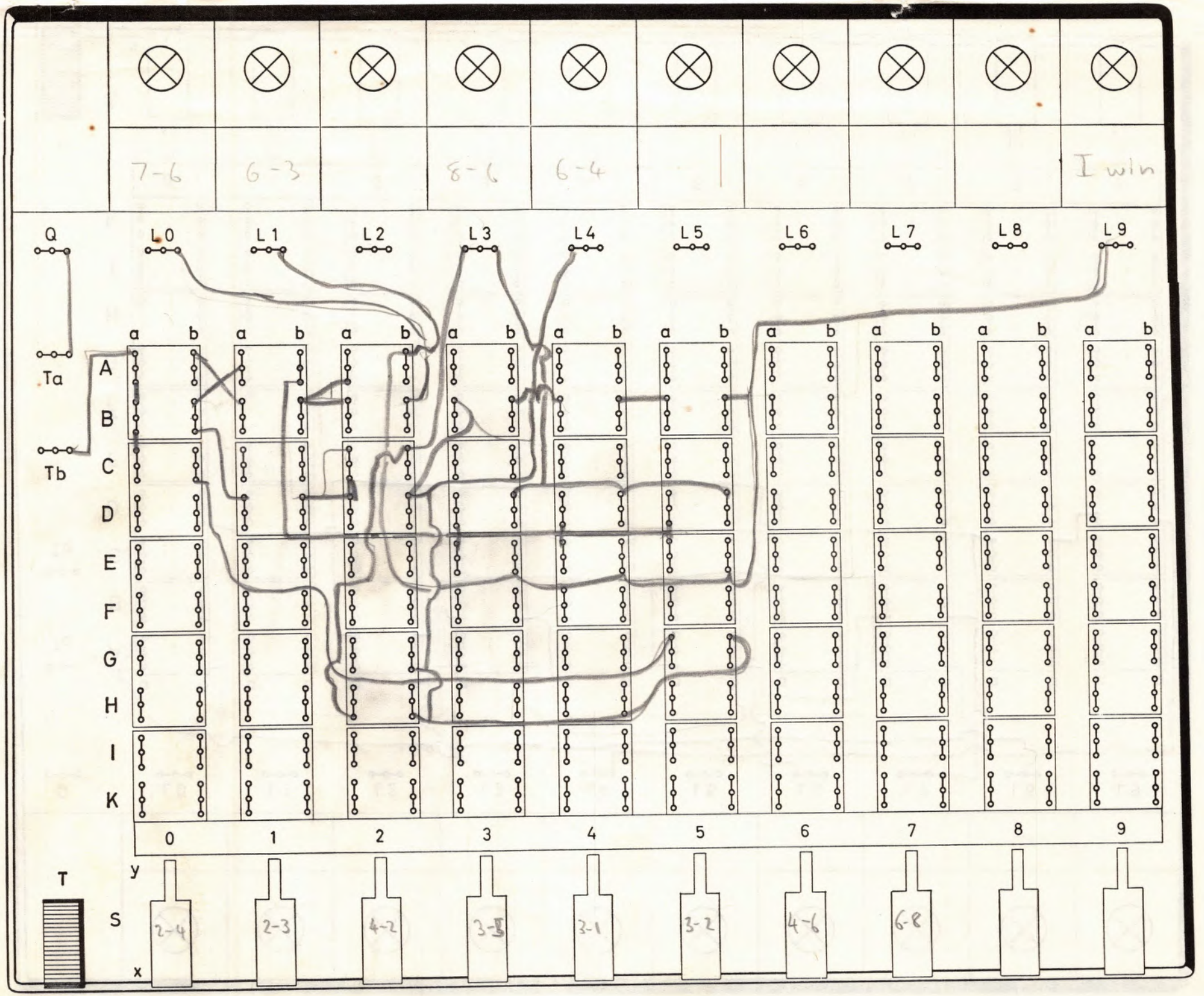
With Program 50, we have come to the end of this instruction manual. This is as far as we go in this book, but you can do much more for yourself.

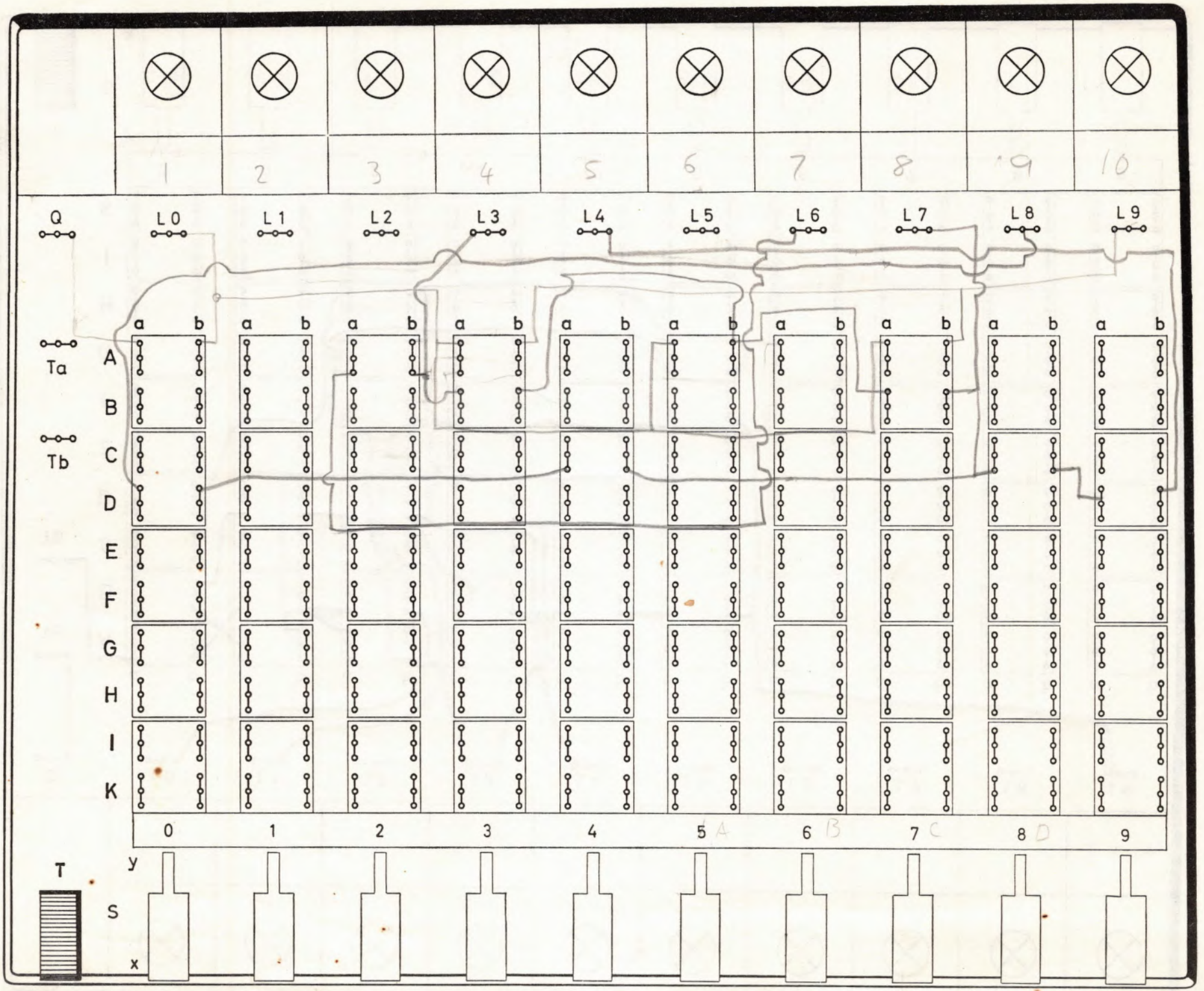
With many experimental kits, the end of the instruction booklet means the end of the experiments; however; your Miniature Computer kit is quite different. This is where you really begin. Now you can attack logical problems on your own, make trial circuits, try out other calculation methods and invent new games. This manual has suggested only a fraction of the possibilities of your computer.

To help you, we have included several blank transparent sheets so that you can write your own inscriptions on them. If you find a unique new program, send us a wiring scheme and description and, if possible, we will include it in our next edition and credit you with its invention.


Q
L 0
L 1
L 2
L 3
L 4
L 5
L 6
L 7
L 8
L 9
Ta
Tb
a
b
A
B
C
D
E
F
G
H
I
K
0
1
2
3
4
5
6
7
8
9
T
y
S
x

L9
L8
L7
L6
L5
L4
L3
L2
L1
L0
Q
Ta
Tb
A
B
C
D
E
F
G
H
I
K
a
b
0
1
2
3
4
5
6
7
8
9
y
S
x
T


7-6
6-3
8-6
6-4
I win
Q
L0
L1
L2
L3
L4
L5
L6
L7
L8
L9
Ta
Tb
A
B
C
D
E
F
G
H
I
K
a
b
0
1
2
3
4
5
6
7
8
9
T
y
S
x
2-4
2-3
4-2
3-5
3-1
3-2
4-6
6-8


1
2
3
4
5
6
7
8
9
10
Q
L0
L1
L2
L3
L4
L5
L6
L7
L8
L9
a
b
Ta
Tb
A
B
C
D
E
F
G
H
I
K
0
1
2
3
4
5
6
7
8
9
A
B
C
D
T
y
S
x

PRINTED IN CANADA

28-199